IMAGE RESTORATION AND RECONSTRUCTION: BEYOND
LEAST SQUARES

A Thesis in
Computer Science and Engineering
by
Haoying Fu

©2005 Haoying Fu

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2005
The thesis of Haoying Fu was reviewed and approved* by the following:

Jesse Barlow
Professor of Computer Science and Engineering
Thesis Adviser
Chair of Committee

Nirmal K. Bose
HRB-Systems Professor of Electrical Engineering

Michael Ng
Associate Professor of Mathematics
University of Hong Kong, Hong Kong

Paul Plassmann
Associate Professor of Computer Science and Engineering

Hongyuan Zha
Associate Professor of Computer Science and Engineering

Raj Acharya
Department Head
Professor of Computer Science and Engineering

*Signatures are on file in the Graduate School.
Abstract

Systems of linear equations with ill conditioned coefficient matrices arise very often in signal and image processing. The most commonly used method for solving such systems is the Regularized Least Squares method, in which the unknown parameters are computed by minimizing a cost function that consists of quadratic data-fitting and regularization terms. We consider techniques other than the Regularized Least Squares for solving such systems. Our focus is on image processing problems.

One implicit assumption behind the least squares solution is that exact information of the coefficient matrix is available. When error exists in both the right hand side and the coefficient matrix, the Total Least Squares method gives better results than the ordinary Least Squares method. We present algorithms for the Regularized Total Least Squares (RTLS) problem.

In many image and signal processing problems, the coefficient matrices have certain useful structure. For example, in the problems of image restoration and high resolution image reconstruction, the resulting blurring matrices have a Block Toeplitz Toeplitz Block (BTTB) like structure. In the problem of color image restoration, the blurring matrix consists of BTTB blocks. However, traditional Total Least Squares methods do not preserve the structure in the coefficient matrix. Therefore it is more appropriate to apply Structured Total Least Squares (STLS). This thesis presents Regularized Structured Total Least Squares (RSTLS) algorithms for the problems of high resolution image reconstruction and color image restoration. The major cost at each iteration of our RSTLS algorithms is in solving large sparse and structured linear least squares systems. We propose to use preconditioned CGLS or LSQR method to solve these systems. We show that Discrete Cosine Transform or Fast Fourier Transform based preconditioners are very effective for these systems.

Other assumptions behind the regularized least squares solution include Gaussian prior distribution of the unknown parameters and the additive noise. When these assumptions are violated, we consider the Least Mixed Norm (LMN) or the Least Absolute Deviation (LAD) solution. For the LAD solution, both the data-fitting and the regularization terms are in the $\ell_1$ norm. For the LMN solution, the regularization term is in the $\ell_1$ norm but the data-fitting term is in the $\ell_2$ norm. Both solutions are formulated as the solution to a convex programming problem, and solved by interior point methods. At each iteration of the interior point method, a structured linear system must be solved. The preconditioned conjugate gradient method with factorized sparse inverse preconditioners is employed to such structured inner systems.
# Table of Contents

List of Figures .................................................. vi

Acknowledgments ................................................ viii

Chapter 1. Introduction ........................................ 1

Chapter 2. Regularized Total Least Squares ....................... 5
  2.1 Preliminaries .............................................. 5
  2.2 Derivation of the algorithm ................................ 6
  2.3 Experimental results ...................................... 10

Chapter 3. Mathematical model for high resolution image reconstruction and color image restoration ........................................ 14
  3.1 Single channel image deblurring ............................ 14
  3.2 High resolution image reconstruction with multi-sensors .... 16
  3.3 Color image restoration .................................... 17

Chapter 4. RSTLS for high resolution image reconstruction and color image restoration ........................................ 19
  4.1 The regularized structured total least squares problem ...... 19
  4.2 RSTLS algorithm for single channel image deblurring ......... 20
    4.2.1 The one dimensional problem ........................... 20
    4.2.2 The two dimensional problem with separable blurring function 23
  4.3 RSTLS algorithm for high resolution image reconstruction .... 28
  4.4 RSTLS for color image restoration .......................... 32
    4.4.1 Regularizing a color image ............................. 32
    4.4.2 Solving the RSTLS problem for color image restoration .. 34
  4.5 Preconditioning the linear least squares systems efficiently ... 38
  4.6 Experimental Results ....................................... 45
    4.6.1 Results on high resolution image reconstruction .......... 46
    4.6.2 Results on color image restoration ..................... 51

Chapter 5. LAD and LMN for image restoration ...................... 58
  5.1 The LAD solution .......................................... 59
    5.1.1 Formulating the problem as a linear programming problem 59
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1.2</td>
<td>Interior point methods</td>
<td>60</td>
</tr>
<tr>
<td>5.1.3</td>
<td>Finding an initial feasible point and staying feasible</td>
<td>63</td>
</tr>
<tr>
<td>5.1.4</td>
<td>Summary of the adapted interior point algorithm</td>
<td>65</td>
</tr>
<tr>
<td>5.2</td>
<td>The LMN solution</td>
<td>65</td>
</tr>
<tr>
<td>5.3</td>
<td>Preconditioning the structured linear systems</td>
<td>68</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Saddle Point Preconditioners</td>
<td>69</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Factorized Sparse Inverse Preconditioner</td>
<td>69</td>
</tr>
<tr>
<td>5.4</td>
<td>Experimental results</td>
<td>73</td>
</tr>
<tr>
<td>6.1</td>
<td>Conclusion</td>
<td>83</td>
</tr>
<tr>
<td>6.2</td>
<td>Future work</td>
<td>84</td>
</tr>
<tr>
<td>6.2.1</td>
<td>Three dimensional image reconstruction</td>
<td>84</td>
</tr>
<tr>
<td>6.2.2</td>
<td>Total least one norm or total least mixed norm for image restoration</td>
<td>85</td>
</tr>
</tbody>
</table>

References | 86 |
# List of Figures

2.1 Convergence of Algorithm 2.1 for the *ilaplace* problem with $m = n = 50$  
2.2 The L-Curve for the *ilaplace* problem with $m = n = 20$, the circle indicates the chosen corner.  
2.3 The true solution, the solution from Algorithm 2.1, the L-curve solution, and the least squares solution for the *ilaplace* problem with $m = n = 20$  

4.1 The original high resolution image  
4.2 The observed high resolution image  
4.3 The L-curve of the RLS and RSTLS method  
4.4 Reconstructed image by the RLS method  
4.5 Reconstructed image by the RSTLS method  
4.6 Error at each CGLS iteration during first outer iteration of Algorithm 4.3, $\alpha = 0.006$  
4.7 The original color image  
4.8 The noisy blurred image  
4.9 The image recovered by the least squares method. The observed image is used to calculate $\beta$. PSNR=27.79  
4.10 The image recovered by the least squares method. The original image is used to calculate $\beta$. PSNR=28.01  
4.11 The image recovered by Algorithm 4.4, PSNR=34.22  
4.12 Error norm at each iteration of Algorithm 4.4  
4.13 The image recovered by a variant of Algorithm 4.4 (a fixed $\beta$ is used), PSNR=33.40  
4.14 The image recovered by a variant of Algorithm 4.4 (without the constraints stated in (4.43)), PSNR=29.47  
4.15 Relative residual for the normal equations at each CGLS iteration during first outer iteration of Algorithm 4.4  

5.1 (a) FSIP with $p = 2$ and $q = 1$; (b) Sparsity pattern of the elements of the coefficient matrix that needs to be calculated.  
5.2 The original image  
5.3 Observed images: (a) all pixels are contaminated by noise; (b) only 50% of the pixels are contaminated by noise.
5.4 (a) The least squares solutions for the first observed image, PSNR = 20.46db, (b) The least squares solution for the second observed image, PSNR = 20.87db 

5.5 The LMN solution for the first observed image, PSNR = 20.78db

5.6 (a) The LMN solution for the second observed image, PSNR = 21.30db, (b) The LAD solution for the second observed image, PSNR = 22.82db

5.7 (a) Original wheel image, (b) Observed noisy image, with 50% of the pixels contaminated by noise, (c) Least Squares restoration, PSNR=29.25 (d) The LMN solution, PSNR=30.64 (e) the LAD solution, PSNR=35.26

5.8 Convergence of the primal-dual interior point method for the LMN solution: (a) Duality measure, (b) Infinity norm of $\Delta f$

5.9 Convergence of the primal-dual interior point method for LAD solution: (a) Duality measure, (b) Infinity norm of $\Delta f$
Acknowledgments

I am most grateful and indebted to my thesis advisor, Professor Jesse Barlow, for the academic and non-academic guidance, patience, and encouragement he has shown me during the past years. I deeply appreciate all the support and guidance from my de facto co-advisor, Dr. Michael Ng. I also sincerely thank Dr. Zha, Dr. Bose, and Dr. Plassmann, for the inspiration, the enlightening discussion, and for being on my committee.

Special thanks for all the people at the Penn State CSE department, and for the University of Hong Kong mathematics department.
Chapter 1

Introduction

We consider the problem of approximately solving an error contaminated linear system of equations \( Ax \approx b \) where \( A \) is \( m \times n \) with \( m \geq n \).

There are many possible ways to find a good solution. One choice, motivated by the Gauss-Markov linear model \([9, \text{Chapter 1}]\), is to let \( x \) be a solution to the minimization problem

\[
\min_{x} \| Ax - b \|_2.
\]  

(1.1)

We call this the least squares approximation of \( b \) from the column space of \( A \).

Our focus is on problems where the coefficient matrix \( A \) is ill conditioned. Such problems arise very often in digital signal and image processing. For these problems, it is necessary to compute a regularized solution since directly solving these systems will yield a solution so sensitive to noise that it is useless.

For the least squares problem, Tikhonov’s regularization \([34]\) is often used to stabilize the solution. A general version of Tikhonov’s regularization takes the form

\[
\min_{x} \| Ax - b \|_2^2 + \alpha \| Rx \|_2^2.
\]  

(1.2)

Here \( \alpha \) is a positive constant chosen to control the size of the solution vector, and \( R \) is a matrix that defines a norm or semi-norm on the solution. We call \( \alpha \) the regularization parameter and \( R \) the regularization operator. The first term in (1.2) is called the data-fitting term and the second term the regularization term. The regularization term can also be interpreted as a quantity that is related to prior knowledge of the distribution of the unknown vector \( x \). In this interpretation, the regularization operator reflects the type of our prior knowledge of the solution. One example is if the solution should be close to some known vector, or smooth. The regularization parameter reflects our confidence in our prior knowledge.

If certain assumptions are satisfied, the solution of (1.2) can be interpreted as the “maximum a posteriori (MAP) estimator” of the unknown vector \( x \). These assumptions include the following:
The vector $b$ is related to the unknown parameter vector $x$ by a linear relation

$$Ax = b + r; \quad (1.3)$$

- the matrix $A$ is known exactly;
- The vector $r$ consists of white Gaussian noise;
- The unknown vector $x$ satisfies a Gaussian prior distribution.

If any one of these assumptions is violated, then (1.2) may yield a solution that is sub-optimal.

For example, in many problems, the coefficient matrix $A$ is also contaminated by noise. In the errors-in-variable model, the following relation is assumed:

$$(A + E)x = b + r. \quad (1.4)$$

For this model, if no regularization is needed, an estimate of the unknown parameters $x$ can be computed by solving the Total Least Squares (TLS) problem

$$\min_{\tilde{A}, x} \| \tilde{A} - A \|_F^2 + \| \tilde{A}x - b \|_2^2. \quad (1.5)$$

Golub et al. [29] showed that Tikhonov’s regularization can be generalized to regularize the TLS solution. The Regularized Total Least Squares problem can be stated as

$$\min_{\tilde{A}, x} \| b - \tilde{A}x \|_2^2 + \| \tilde{A} - A \|_F^2 + \alpha \| Rx \|_2^2. \quad (1.6)$$

In many signal and image processing problems, the coefficient matrix exhibits a certain structure. For example, for the one dimensional single channel image deblurring problem, the resulting blurring matrix is a Toeplitz matrix when zero boundary conditions are used. An $m \times n$ matrix $T = (t_{kj})$ is Toeplitz if it has constant entries along each diagonal. That is, a Toeplitz matrix has the form

$$T = \begin{bmatrix} t_0 & t_1 & t_2 & \cdots & t_{n-1} \\
    t_{-1} & t_0 & t_1 & \cdots & : \\
    t_{-2} & t_{-1} & t_0 & \cdots & : \\
    : & \ddots & \ddots & \ddots & \vdots \\
    t_{1-m} & \cdots & \cdots & t_{n-m-1} & t_{n-m} \end{bmatrix}.$$
In this thesis, we use the MATLAB [26] notation $\text{Toeplitz}(c, r)$ to denote the Toeplitz matrix having $c$ as its first column and $r$ as its first row. When Neumann boundary conditions are used, the resulting blurring matrix is a Toeplitz plus Hankel matrix in the one dimensional single channel case. A matrix is called a Hankel matrix if it has constant entries along its anti-diagonals.

For two dimensional single channel image deblurring problems, the resulting blurring matrix is Block Toeplitz with Toeplitz Blocks (BTTB) when zero boundary conditions are used. It is Block Toeplitz plus Hankel with Toeplitz plus Hankel Blocks (BTHTHB) when Neumann boundary conditions are used.

The problem of reconstructing a high resolution image from multiple under-sampled frames [13] is closely related to the problem of image deblurring. Multiple under-sampled image frames are often obtained by using identical image sensors shifted from each other by sub-pixel displacements. When the under-sampled frames are shifted from each other by exact sub-pixel displacements, the problem of high resolution image reconstruction becomes a single image deblurring problem. However, in practice, the sub-pixel displacements are seldom perfect. This causes the resulting discretization matrix to be a “BTTB-like” matrix, but not exactly BTTB when zero boundary conditions are used. It is “BTHTHB-like” but not exactly BTHTHB when Neumann boundary conditions are used.

A color image may be regarded as a set of three images in the three color components, red, green and blue. A more general term, multichannel images, is often used to denote color images. In that case, each color component is considered as a channel. For the problem of color image restoration, the blurring matrix consists of $k \times k$ blocks, where each block is a single channel blurring matrix, and $k$ is the number of channels.

Traditional TLS methods are based on the singular value decomposition. They do not preserve the structure of the coefficient matrix when applied to these problems. Recently there has been considerable research effort in applying the Structured Total Least Squares (STLS) method to the problems of image deblurring and high resolution image reconstruction. Rosen et al. [59] presented a STLS solution for simple Toeplitz matrices. Using STLS with Tikhonov regularization for image deblurring has been previously studied in [43], [50], and [56]. Regularized STLS (RSTLS) algorithm for high resolution image reconstruction has been recently considered by Ng et al. in [48]. In this thesis, we present a RSTLS algorithm for high resolution image reconstruction. The new method is faster and more accurate than the technique described in [48].

On the problem of color image restoration, there is considerable work in the literature on least squares solutions. Based on the assumption that the within-channel
and cross-channel point spread functions are separable, Hunt and Kubler [35] derived a minimum MSE multichannel restoration filter. Filters that utilize the cross channel correlation without using the separability assumption can be found in [27] and [28]. More results can be found in [46, 49, 66]. However, no results have been reported on using TLS for color image restoration. We consider applying RSTLS to this problem.

Another assumption that must be satisfied in order for (1.2) to yield the MAP estimator and is often violated is that the unknown parameters follow a Gaussian prior distribution. For example, if the sought after vector $x$ represents an image, then its prior distribution rarely satisfies the Gaussian assumption well due to the presence of information bearing edges. In this thesis, we consider the Laplace prior distribution [5].

Using the Laplace prior distribution assumption leads to the use of $\ell_1$ norm rather than the $\ell_2$ norm for the regularization term:

$$\min_f \| \mathbf{b} - \mathbf{A}x \|_2^2 + \alpha \| Rx \|_1.$$  (1.7)

We call the solution to (1.7) the least mixed norm (LMN) solution. We remark that in the literature, there has been a growing interest in using $\ell_1$ norm for parameter estimation [6, 5, 4, 36, 11, 14, 61], and for image restoration [3, 17, 31, 38, 44, 51, 52]. The advantage of using the $\ell_1$ norm is that the solution is more robust than $\ell_2$ norm in statistical estimation problems. In particular, a small number of outliers usually do not change the solution much, see for instance [11, 51].

It is quite possible that the the additive noise does not satisfy the Gaussian assumption either. In that case, we consider using $\ell_1$ norm for both the data-fitting and the regularization terms:

$$\min_x \| \mathbf{b} - \mathbf{A}x \|_1 + \alpha \| Rx \|_1.$$  (1.8)

We call the solution to (1.8) the least absolute deviation (LAD) solution. Recently, Nikolova [51, 52], Chan and Esedoglu [17] have studied minimizers of cost-functions involving $\ell_1$ data-fidelity.

The rest of this thesis is organized as follows. In Chapter 2, we present an algorithm for calculating the regularization parameters for the Regularized Total Least Squares (RTLS) problem. In Chapter 3, we give a brief description of the mathematical model for the image processing problems mentioned above. Chapter 4 presents the RSTLS algorithm for high resolution image reconstruction and color image restoration. Chapter 5 presents the LMN and LAD algorithm for image restoration and high resolution image reconstruction. Chapter 6 concludes this thesis and suggests directions for future research.
Chapter 2

Regularized Total Least Squares

2.1 Preliminaries

Recall that the RTLS problem can be stated as

\[
\min_{\tilde{A}, x} (\|b - \tilde{A}x\|^2_2 + \|\tilde{A} - A\|_F^2 + \alpha\|Rx\|^2_2).
\]  

(2.1)

Usually the regularization parameter \(\alpha\) is not known \textit{a priori}, depending on the application, several methods can be used to choose an appropriate regularization parameter.

In many applications, it is desired that the solution solves the following problem:

\[
\min_{\tilde{A}, x} (\|b - \tilde{A}x\|^2_2 + \|\tilde{A} - A\|_F^2) \text{ subject to } \|Rx\|^2_2 \leq \delta^2.
\]  

(2.2)

Here the parameter \(\delta\) comes from knowledge of the underlying physical model. The Lagrange multiplier formulation for (2.2) is

\[
\mathcal{L}(\tilde{A}, x, \mu) = \|b - \tilde{A}x\|^2_2 + \|\tilde{A} - A\|_F^2 + \mu(\|Rx\|^2_2 - \delta^2),
\]  

(2.3)

where \(\mu\) is the Lagrange multiplier. It can be shown [29] that if \(\delta \leq \|Rx_{TLS}\|_2\), then the solution to (2.2) is identical to that of (2.1) for appropriately chosen \(\alpha\); conversely, the solution of (2.1) is identical to that of (2.2) for appropriately chosen \(\delta\).

Another widely used method for choosing the optimal regularization parameter is the L-Curve method [34]. The L-curve is a plot of the size of the regularized solution versus the norm of the corresponding residual, it exhibits a corner behavior as a function of the regularization parameter \(\alpha\) (Figure 2.2). The regularization parameter is often chosen to be on the corner of the L-curve, because the regularized solution on the corner yields both a small residual norm and a small solution size.

Golub et al. [29] showed that if the constraint is active, the solution to (2.2) is a solution to the problem

\[
(A^TA + \alpha_I I_n + \alpha_L R^TR)x = A^Tb,
\]  

(2.4)
where
\[ \alpha_I = -\frac{\|Ax - b\|^2}{1 + \|x\|^2}, \quad (2.5) \]
\[ \alpha_L = \mu(1 + \|x\|^2). \quad (2.6) \]

Here \( \mu \) is the Lagrange multiplier in (2.3). This result was extended by Guo and Renaut [32], who showed that the solution to (2.2) satisfies
\[
\begin{bmatrix}
A^T A + \alpha_L R^T R & A^T b \\
-\alpha_L \delta^2 + b^T b & -1
\end{bmatrix}
\begin{bmatrix}
x \\
-1
\end{bmatrix}
= -\alpha_I
\begin{bmatrix}
x \\
-1
\end{bmatrix}.
\quad (2.7)
\]

Guo and Renaut then used an iterative algorithm that resembles the RQI to find the parameters \( \alpha_L \) and \( \alpha_I \). Sima et al. [64] proposed solving the problem by an iterative procedure based on the quadratic eigenvalue problem. Later Renaut and Guo [58] improved their earlier algorithm, and claimed that their algorithm outperforms the algorithm by Sima et al. They also presented algorithms based on bisection search and the L-curve approach in this later paper.

### 2.2 Derivation of the algorithm

Define functions \( \phi_1(\alpha_I, \alpha_L) \) and \( \phi_2(\alpha_I, \alpha_L) \) as the following:
\[
\phi_1(\alpha_I, \alpha_L) = \frac{1}{\delta} - \frac{1}{\|Rx(\alpha_I, \alpha_L)\|_2}, \quad (2.8)
\]
\[
\phi_2(\alpha_I, \alpha_L) = -\alpha_I - \frac{\|Ax(\alpha_I, \alpha_L) - b\|^2}{1 + \|x(\alpha_I, \alpha_L)\|^2}. \quad (2.9)
\]

Then the parameters \( \alpha_I \) and \( \alpha_L \) satisfy the system of nonlinear equations:
\[
\begin{cases}
\phi_1(\alpha_I, \alpha_L) = 0 \\
\phi_2(\alpha_I, \alpha_L) = 0
\end{cases}.
\quad (2.10)
\]

This is a simple nonlinear system of two variables. Since it is rather difficult to compute the analytical Jacobian matrix of (2.10), we use secant updating methods to solve this system. One of the simplest and most effective secant updating methods for systems of nonlinear equations is the Broyden’s method [53, §11.1].

We remark that for the function \( \phi_1 \), the more obvious choice would be the following:
\[
\hat{\phi}_1(\alpha_I, \alpha_L) = \delta - \|Rx(\alpha_I, \alpha_L)\|_2.
\]
However, as was pointed out in [53, page 81], functions similar to $\phi_1$ are highly nonlinear for values close to the solution, therefore root-finding Newton or quasi-Newton methods are slow. Better results will be obtained if (2.8) is used because it is nearly linear near the solution.

Let $\alpha$ denote the vector $[\alpha_I \, \alpha_L]^T$ and $\phi$ denote the vector $[\phi_1 \, \phi_2]^T$, where $\phi_1$ and $\phi_2$ are defined in (2.8) and (2.9), then Algorithm 2.1 summarizes the adapted Broyden’s method for solving (2.10).

\begin{algorithm}
\begin{enumerate}
  \item initialize $\alpha_L$, $\alpha_I$, compute $x(\alpha_L, \alpha_I)$.
  \item calculate $\phi^{(0)}$. Use finite differences to calculate an approximate initial Jacobian $J^{(0)}$.
  \item while $\|\phi^{(k)}\|_2 > tol$
    \begin{enumerate}
      \item Solve $J^{(k)} \Delta \alpha = -\phi^{(k)}$
      \item $\alpha^{(k+1)} = \alpha^{(k)} + \Delta \alpha$
      \item Solve (2.4), calculate $\phi^{(k+1)}$
      \item Use the Broyden update formula to update $J$:
        \begin{equation}
          J^{(k+1)} = J^{(k)} + \left( \phi^{(k+1)} - \phi^{(k)} - J^{(k)} \Delta \alpha \right) \Delta \alpha^T / (\Delta \alpha^T \Delta \alpha)
        \end{equation}
      \item $k = k + 1$
    \end{enumerate}
\end{enumerate}
\end{algorithm}

It is clear that the major work at each iteration of Algorithm 2.1 is solving a linear system in the form of (2.4).

If the initial guess is close to the solution, Broyden’s method converges super-linearly. One apparent choice for the initial guess would be to choose $\alpha_L$ to be the least squares regularization parameter and $\alpha_I$ be zero. Also experimental results indicate that for problems of reasonable noise level, an initial guess of $\alpha_L = 0.1$ and $\alpha_I = 0$ is good enough. A more robust approach for finding a good initial guess will be explained later.

In practical implementations, we require that $\alpha_L$ be non-negative and $\alpha_I$ be non-positive. At any iteration, if taking a full step will violate those conditions, we reduce the step length to $0.95 \beta_{max}$, where $\beta_{max}$ is the maximum step length that can be taken before those conditions are violated. This works well for all the experiments we carried out. A more sophisticated approach would be to use a step length that leads to sufficient decrease of $\|\phi\|_2$. 
In situations where $\delta$ is not given, the L-curve method is often used to determine the regularization parameters. An algorithm for plotting the L-Curve was given in [58] by Renaut and Guo. They proposed using a fixed point type iteration to solve (2.5) as an equation of $\alpha_I$ alone. Here we propose an algorithm that we claim is more robust. Before we do that, we prove the following lemma first.

**Lemma 2.1.** For a given $\alpha_L$, let $\alpha_I$ be such that $(\alpha_L, \alpha_I)$ solves (2.2) for an appropriate choice of $\delta$, let $M = A^T A + \alpha_L R^T R + \alpha_I I_n$. Then $\alpha_I$ is the root of smallest magnitude of the secular equation:

$$f(\alpha_I) = b^T b - \alpha_L b^T A M^{-1} R^T R M^{-1} A^T b + \alpha_I - b^T A M^{-1} A^T b = 0.$$  \hspace{1cm} (2.11)

**Proof.** Let $\alpha_I^*$ be the root of smallest magnitude of (2.11), let $x$ be the vector that solves (2.4) for $\alpha_I = \alpha_I^*$. Then $\alpha_I^*$ and $x$ satisfy

$$\begin{bmatrix} A^T A + \alpha_L R^T R \\ b^T A \\ -\alpha_L b^T R^T R M^{-1} A^T b - \alpha_I I \\ b^T b \end{bmatrix} x = \begin{bmatrix} x \\ -1 \\ -1 \end{bmatrix} = \begin{bmatrix} \alpha_I^* \\ \alpha_I^* \\ x \\ -1 \end{bmatrix}. \hspace{1cm} (2.12)$$

By comparing (2.12) and (2.7), we see that the only difference is that the $\delta$ in (2.7) is replaced by $\|R x\|_2$ in (2.12). Therefore, $(\alpha_L, \alpha_I^*)$ solves (2.2) for $\delta = \|R x\|_2$. \qed

The secular equation (2.11) can be solved by either Newton’s method or the secant method. For the secant method, the function $f(\alpha_I)$ needs to be evaluated at each iteration, which requires solving a linear system of the form $M x = c$. In [29], Golub et al. showed how to solve this system efficiently for many values of $\alpha_I$. Another technique that can be used here is the one described in [25]. Let

$$\begin{bmatrix} A \\ \sqrt{\alpha_L} R \\ \sqrt{\alpha_L} R^T R \end{bmatrix} = U B V^T \hspace{1cm} (2.13)$$

be the bidiagonalization of the matrix $[A^T \sqrt{\alpha_L} R^T]^T$, then $A^T A + \alpha_L R^T R = V B^T B V^T$. Therefore, we have

$$A^T A + \alpha_L R^T R + \alpha_I I = V (B^T B + \alpha_I I) V^T.$$  

By using the technique in [25], we can easily find a bidiagonal matrix $\hat{B}$ and an orthogonal matrix $\hat{V}$ such that

$$\hat{V} \hat{B}^T \hat{B} \hat{V}^T = B^T B + \alpha_I I.$$ \hspace{1cm} (2.14)
Therefore, for each fixed value of $\alpha_L$, only one bidiagonalization of the matrix $[A^T \sqrt{\alpha_L R^T}]^T$ is needed. Once we have (2.13), the linear system of the form $Mx = c$ can be easily solved, and $f(\alpha_I)$ can be easily evaluated.

Newton’s method requires the evaluation of the derivative of $f(\alpha_I)$, which is given by

$$f'(\alpha_I) = 2\alpha_L b^T A M^{-2} R^T R M^{-1} A^T b + 1 + b^T A M^{-2} A^T b.$$  

(2.15)

It is clear that evaluating the derivative requires solving a linear system of the form $Mx = c$ for two different right hand sides. If we use the technique mentioned above, solving the system for two different right hand sides requires about twice as much work as solving it once. Therefore, secant method is preferred over Newton’s method. The secant method needs two initial points for $\alpha_I$, we can use zero for one of the initial points, and the other one can be found by performing one step of Newton’s method. Algorithm 2.2 summarizes the procedure for finding the corresponding $\alpha_I$ and $\delta$ for any given $\alpha_L$.

---

**Algorithm 2.2**

1. find the bidiagonalization of $[A^T \sqrt{\alpha_L R^T}]^T$.
2. solve $(A^T A + \alpha_L R^T R)\tilde{x} = A^T b$ and $(A^T A + \alpha_L R^T R)x = \tilde{x}$.
   use $\tilde{x}$ and $x$ to evaluate $f(0)$ and $f'(0)$ according to (2.11) and (2.15), $\alpha_{I1} = 0, \alpha_{I2} = -f(0)/f'(0)$
3. while $f(\alpha_{I2}) > tol$
   solve $(A^T A + \alpha_L R^T R + \alpha_{I2} I)x = A^T b$
   evaluate $f(\alpha_{I2})$
   $h = -f(\alpha_{I2})(\alpha_{I2} - \alpha_{I1})/(f(\alpha_{I2}) - f(\alpha_{I1}))$
   $\alpha_{I1} = \alpha_{I2}, \alpha_{I2} = \alpha_{I2} + h$
   end
4. $\alpha_I = \alpha_{I2}$
   $\delta = \|Rx\|$  

---

To plot the L-Curve, we simply try different values of $\alpha_L$ and use Algorithm 2.2 to find the corresponding $\alpha_I$ and $\delta$ (which is exactly the solution size). Based on the $(\alpha_I, \delta)$ pairs, we can plot the L-curve.

Algorithm 2.2 can also be used to find a good initial guess of $\alpha_L$ and $\alpha_I$ for Algorithm 2.1. Since for any given $\alpha_L$, its corresponding $\alpha_I$ can be found by Algorithm 2.2, we need only an initial guess of $\alpha_L$ to start the Broyden’s method. We suggest that one start with $\alpha_L = 1$ and decrease $\alpha_L$ by a factor of 10 repeatedly until a $\alpha_L$ whose corresponding $\delta$(which is also found by Algorithm 2.2) is bigger than the required $\delta$ is found.
2.3 Experimental results

Three problems, \textit{ilaplace}, \textit{deriv2}, and \textit{baart}, from Hansen’s “Regularization Tools” [33] were used to test our algorithms. The functions from Hansen’s toolbox return a square matrix \( A \), the true solution \( \mathbf{x} \), and a right hand side \( \mathbf{b} \) such that \( A \mathbf{x} = \mathbf{b} \) is satisfied. The size of the test problems we used ranges from 20 to 500, with the exception of the \textit{ilaplace} problem, because the \textit{ilaplace} function from Hansen’s toolbox returns a coefficient matrix with “NaN” entries when the size of the problem exceeds 200. For the purpose of testing our algorithms for rectangular systems, the \textit{ilaplace} function was slightly modified such that it can return a rectangular matrix. In all cases the coefficient matrix and the right hand side are scaled such that \( \| A \|_F \) and \( \| \mathbf{b} \|_2 \) are both equal to one, and the true solution is adjusted correspondingly.

An error matrix \( E \) was generated by multiplying each element of the matrix \( A \) by a random number, generated by the MATLAB [26] “randn” function. The error contaminated coefficient matrix, \( A_E \), was generated as follows:

\[
A_E = A + \sigma \|E\|^{-1}_F E,
\]

where \( \sigma \) represents the noise level. The noisy right hand side, \( \mathbf{b}_E \), was generated as follows:

\[
\mathbf{b}_E = \mathbf{b} + \sigma \| \mathbf{e} \|_2^{-1} \mathbf{e},
\]

where the elements of the vector \( \mathbf{e} \) come from the MATLAB “randn” function.

The regularization operator \( R \) is set to be the first order difference operator. In these tests, since the true solution is known, we set \( \delta \) to \( \| R \mathbf{x} \|_2 \). For all tests carried out here, the initial guess for \( \alpha_L \) and \( \alpha_I \) is set to be 0.1 and 0 respectively.

Table 2.1 shows the number of iterations needed by Algorithm 2.1 to converge for different tests. As can be seen, this algorithm converges very quickly for all the tests carried out here. The results reported in [58] indicate that their algorithm needed considerably more iterations to converge. Note that for both algorithms, the major cost of each iteration is solving a linear system in the form of (2.4). The calculated \( \alpha_L \) and \( \alpha_I \) are also shown in Table 2.1. Note that in all experiments, the magnitude of \( \alpha_I \) is small enough to indicate a good fit, but big enough to justify that the TLS solution is considerably different from the least squares solution. For the problems of \textit{deriv2} and \textit{baart}, \( \alpha_L \) seems to increase with the size of the problem, this is because the size of the true solution returned by these functions decreases when the size of these problem increases.

Figure 2.1 shows the convergence of Algorithm 2.1 for the \textit{ilaplace} problem with \( m = n = 50 \), the general trend indicates a super linear convergence once the iterates are close to the solution. Figure 2.2 shows the L-curve for the \textit{ilaplace} problem with \( m = n = 20 \). According to the L-curve, the optimal \( \alpha_L \) is 8.0e-2. The true solution, the solution from Algorithm 2.1, the L-curve solution, and the least squares solution for this problem are shown in Figure 2.3.
Table 2.1. Number of iterations needed by Algorithm 2.1 to converge

<table>
<thead>
<tr>
<th>Algorithm, $\sigma = 0.05$</th>
<th>No. Iter</th>
<th>$\alpha_L$</th>
<th>$\alpha_I$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ilaplace, $\sigma = 0.05$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$m=n=20$</td>
<td>9</td>
<td>7.99e-2</td>
<td>-2.07e-4</td>
</tr>
<tr>
<td>$m=40, n=20$</td>
<td>16</td>
<td>3.95e-2</td>
<td>-1.95e-4</td>
</tr>
<tr>
<td>$m=n=50$</td>
<td>20</td>
<td>2.83e-1</td>
<td>-2.02e-4</td>
</tr>
<tr>
<td>$m=100, n=50$</td>
<td>16</td>
<td>2.50e-1</td>
<td>-2.91e-4</td>
</tr>
<tr>
<td>$m=n=100$</td>
<td>12</td>
<td>8.32e-1</td>
<td>-1.70e-4</td>
</tr>
<tr>
<td>$m=200, n=100$</td>
<td>15</td>
<td>7.70e-1</td>
<td>-2.67e-4</td>
</tr>
<tr>
<td>deriv2, $\sigma = 0.05$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$m=n=20$</td>
<td>11</td>
<td>8.10e-3</td>
<td>-4.89e-4</td>
</tr>
<tr>
<td>$m=n=50$</td>
<td>7</td>
<td>3.18e-1</td>
<td>-8.54e-4</td>
</tr>
<tr>
<td>$m=n=100$</td>
<td>11</td>
<td>7.75e-1</td>
<td>-9.19e-4</td>
</tr>
<tr>
<td>$m=n=200$</td>
<td>14</td>
<td>3.49e+0</td>
<td>-9.25e-4</td>
</tr>
<tr>
<td>$m=n=500$</td>
<td>16</td>
<td>1.58e+1</td>
<td>-9.15e-4</td>
</tr>
<tr>
<td>baart, $\sigma = 0.05$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$m=n=20$</td>
<td>15</td>
<td>8.83e-3</td>
<td>-8.66e-4</td>
</tr>
<tr>
<td>$m=n=50$</td>
<td>13</td>
<td>6.48e-2</td>
<td>-7.94e-4</td>
</tr>
<tr>
<td>$m=n=100$</td>
<td>9</td>
<td>2.36e-1</td>
<td>-7.97e-4</td>
</tr>
<tr>
<td>$m=n=200$</td>
<td>8</td>
<td>9.45e-1</td>
<td>-8.03e-4</td>
</tr>
<tr>
<td>$m=n=500$</td>
<td>16</td>
<td>4.17e+0</td>
<td>-8.02e-4</td>
</tr>
</tbody>
</table>
Fig. 2.1. Convergence of Algorithm 2.1 for the *ilaplace* problem with $m = n = 50$

Fig. 2.2. The L-Curve for the *ilaplace* problem with $m = n = 20$, the circle indicates the chosen corner.
Fig. 2.3. The true solution, the solution from Algorithm 2.1, the L-curve solution, and the least squares solution for the *ilaplace* problem with $m = n = 20$
3.1 Single channel image deblurring

For the one dimensional single channel deblurring problem [30, §5.2], let the original image or signal be $f(x)$, the blurring function be $h(x)$, and the noise present in the observed image be $\eta(x)$, then the observed image $g(x)$ is given by

$$g(x) = h(x) \ast f(x) + \eta(x).$$

(3.1)

Here $\ast$ denotes convolution. The continuous image model can be discretized by the Trapezoid rule and approximated by a discrete image. Let the discretized original scene be $[\ldots, f_{-m+1}, \ldots, f_0, f_1, \ldots, f_n, f_{n+1}, \ldots, f_{n+m}, \ldots]^T$, the discretized blurring function be $[\ldots, 0, 0, h_{-m}, h_{-m+1}, \ldots, h_0, \ldots, h_{m-1}, h_m, 0, 0, \ldots]^T$, and the discretized noise be $[\ldots, \eta_{-m+1}, \ldots, \eta_0, \eta_1, \ldots, \eta_n, \eta_{n+1}, \ldots, \eta_{n+m}, \ldots]^T$. Then the $i$th entry of the discretized blurred image or signal is given by

$$g_i = \sum_{j=1-m}^{i+m} h_{i-j} f_j + \eta_i \quad \text{for } i = 1, 2, \ldots, n.$$ 

(3.2)

Assuming $m < n$, the above system can be written in a matrix-vector form

$$T_l f_l + T f + T_r f_r + \eta = g,$$

(3.3)

where the terms $T_l f_l$ and $T_r f_r$ denote boundary information. Here

$$T_l = \text{Toeplitz}([0, \ldots, 0]^T, [0, \ldots, 0, h_m, h_{m-1}, \ldots, h_1]^T),$$

$$f_l = [0, \ldots, 0, f_{-m+1}, f_{-m+2}, \ldots, f_{-1}, f_0]^T,$$

$$T = \text{Toeplitz}([h_0, h_1, \ldots, h_m, 0, \ldots, 0]^T, [h_0, h_{-1}, \ldots, h_{-m}, 0, \ldots, 0]^T),$$

$$f = [f_1, f_2, \ldots, f_n]^T,$$
\[ T_r = \text{Toeplitz}([0, \ldots, 0, h_{-m}, h_{-m+1}, \ldots, h_{-1}])^T, [0, \ldots, 0]^T, \]
\[ f_r = [f_{n+1}, f_{n+2}, \ldots, f_{n+m}, 0, \ldots, 0]^T, \]
\[ \eta = [\eta_1, \eta_2, \ldots, \eta_n]^T, \]
and
\[ g = [g_1, g_2, \ldots, g_n]^T. \]

The linear system (3.3) is an under-determined system, certain assumptions on the boundary conditions are needed to reduce the number of unknowns.

The zero boundary condition assumes that the image outside the domain of the observed image is zero (totally dark), that is,
\[ f_l = f_r = 0. \]

Then equation (3.3) becomes
\[ Hf + \eta = g \quad \text{where} \quad H = T. \quad (3.4) \]

Therefore \( H \) is a Toeplitz matrix of bandwidth \( m \).

The Neumann boundary condition assumes that the image outside the domain of the image is a reflection of the image inside, or
\[
\begin{align*}
\begin{cases}
  f_0 & = f_1 \\
  \vdots & = \vdots \\
  f_{-m+1} & = f_m \\
\end{cases}
\quad \text{and} \quad 
\begin{cases}
  f_{n+1} & = f_n \\
  \vdots & = \vdots \\
  f_{n+m} & = f_{n-m+1} \\
\end{cases}
\end{align*}
\]

Then equation (3.3) becomes
\[ Hf + \eta = g \quad \text{where} \quad H = T_l J + T + T_r J. \quad (3.5) \]

Here \( J \) is the \( n \times n \) exchange matrix. It is clear that \( H \) is a Toeplitz plus Hankel matrix of bandwidth \( m \).

For two dimensional problems, the resulting blurring matrix is a BTTB matrix under zero boundary conditions, and BTHTHB matrix under Neumann boundary conditions. In the case that the blurring function is separable, that is, it can be expressed as the product of two one dimensional blurring functions, the blurring matrix is also separable. In other words, the blurring matrix has the form of \( H_x \otimes H_y \), where \( H_x \) and \( H_y \) are one dimensional blurring matrices, and \( \otimes \) denotes the Kronecker product.
### 3.2 High resolution image reconstruction with multi-sensors

A brief description of the mathematical model from [13] for high-resolution image reconstruction is given below.

Consider a sensor array with $L_1 \times L_2$ sensors in which each sensor has $N_1 \times N_2$ sensing elements (pixels) and the size of each sensing element is $P \times Q$. Our goal is to reconstruct an image of resolution $M_1 \times M_2$, where $M_1 = L_1 N_1$ and $M_2 = L_2 N_2$. The pixel size of the reconstructed image is $P/L_1 \times Q/L_2$. To maintain the aspect ratio of the reconstructed image, only the case where $L_1 = L_2 = L$ is considered in this research.

To generate enough information to solve for the high-resolution image, sub-pixel displacements between the sensors are necessary. Ideally, the sensors are shifted from each other by a multiple of the high-resolution pixel size $P/L_1 \times Q/L_2$. In practice, there will be small perturbations around the ideal sub-pixel displacements. Therefore, for $\ell_1, \ell_2 = 0, 1, \ldots, L-1$ with $(\ell_1, \ell_2) \neq (0, 0)$, the horizontal and vertical displacements $d^x_{\ell_1, \ell_2}$ and $d^y_{\ell_1, \ell_2}$ can be modeled as

$$
\begin{align*}
    d^x_{\ell_1, \ell_2} &= \frac{P}{L}(\ell_1 + \epsilon^x_{\ell_1, \ell_2}) \quad \text{and} \quad d^y_{\ell_1, \ell_2} = \frac{Q}{L}(\ell_2 + \epsilon^y_{\ell_1, \ell_2})
\end{align*}
$$

where $\epsilon^x_{\ell_1, \ell_2}$ and $\epsilon^y_{\ell_1, \ell_2}$ denote the normalized differences between the actual and ideal displacements in the $x$ and $y$ direction respectively. It is assumed that

$$
|\epsilon^x_{\ell_1, \ell_2}|, |\epsilon^y_{\ell_1, \ell_2}| < \frac{1}{2}.
$$

Let $f(x, y)$ be the original scene. The low resolution image obtained from the $(\ell_1, \ell_2)$-th sensor, $g_{\ell_1, \ell_2}$, is modeled by

$$
\begin{align*}
ge_{\ell_1, \ell_2}(n_1, n_2) &= \int_{Q(n_2-1/2)+d^y_{\ell_1, \ell_2}}^{Q(n_2+1/2)+d^y_{\ell_1, \ell_2}} \int_{P(n_1-1/2)+d^x_{\ell_1, \ell_2}}^{P(n_1+1/2)+d^x_{\ell_1, \ell_2}} f(x, y) \, dx \, dy + \eta_{\ell_1, \ell_2}(n_1, n_2).
\end{align*}
$$

By a discretization of the continuous image model, we obtain the blurring matrix corresponding to the $(\ell_1, \ell_2)$-th sensor, which is given by

$$
H_{\ell_1, \ell_2} = H^x_{\ell_1, \ell_2} \otimes H^y_{\ell_1, \ell_2}.
$$

Here $H^x_{\ell_1, \ell_2}$ is the one dimensional blurring matrix of size $M_1 \times M_1$ defined as

$$
H^x_{\ell_1, \ell_2} = \begin{cases}
T & \text{under zero boundary conditions} \\
T_{\ell} J + T + T_{\gamma} J & \text{under Neumann boundary conditions}
\end{cases}
$$

(3.6)
where

\[ T = \frac{1}{L} \cdot \text{Toeplitz}([1, \ldots, 1, \frac{1}{2} + \epsilon_{\ell_1, \ell_2}, 0, \ldots, 0]^T, [1, \ldots, 1, \frac{1}{2} - \epsilon_{\ell_1, \ell_2}, 0, \ldots, 0]^T), \]

\[ T_\ell = \frac{1}{L} \cdot \text{Toeplitz}([0, \ldots, 0]^T, [0, \ldots, 0, \frac{1}{2} + \epsilon_{\ell_1, \ell_2}, 1, \ldots, 1]^T), \]

\[ T_r = \frac{1}{L} \cdot \text{Toeplitz}([0, \ldots, 0, \frac{1}{2} - \epsilon_{\ell_1, \ell_2}, 1, \ldots, 1]^T, [0, \ldots, 0]^T). \]

\( H_{\ell_1, \ell_2}^y \) is the \( M_2 \times M_2 \) one dimensional blurring matrix defined similarly. We see that the blurring matrix of each sensor has a separable form.

We have now established that \( H_{\ell_1, \ell_2}^x \) and \( H_{\ell_1, \ell_2}^y \) are Toeplitz matrices of bandwidth \( L/2 \) under zero boundary conditions, and that they are Toeplitz plus Hankel matrices of bandwidth \( L/2 \) under Neumann boundary conditions.

Gathering the low resolution frames, we get

\[ Hf + \eta = g, \] (3.7)

where

\[ H = \sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1, \ell_2} H_{\ell_1, \ell_2}^x \otimes H_{\ell_1, \ell_2}^y. \]

Here \( D_{\ell_1, \ell_2} \) are diagonal matrices whose diagonal elements are equal to one if the corresponding element of \( g \) comes from the \((\ell_1, \ell_2)\)-th sensor and zero otherwise.

### 3.3 Color image restoration

Let \( f_r(x,y), f_g(x,y) \) and \( f_b(x,y) \) be the original red, green and blue scenes. Let \( u_r(x,y), u_g(x,y) \) and \( u_b(x,y) \) be the observed red, green and blue scenes. Let \( h_{ij}(x,y) \) (for \( i,j \in \{r,g,b\} \)) be the within-channel and cross-channel point spread functions, normalized such that each function integrates to one. Let \( \eta_r(x,y), \eta_g(x,y) \) and \( \eta_b(x,y) \) be the noise present in the observed image. The image formation process can be modeled as follows [27, 28]:

\[ u_r = w_{rr} h_{rr} \ast f_r + w_{rg} h_{gr} \ast f_g + w_{br} h_{br} \ast f_b + \eta_r, \]

\[ u_g = w_{rg} h_{rg} \ast f_r + w_{gg} h_{gg} \ast f_g + w_{bg} h_{bg} \ast f_b + \eta_g, \]

\[ u_b = w_{rb} h_{rb} \ast f_r + w_{gb} h_{gb} \ast f_g + w_{bb} h_{bb} \ast f_b + \eta_b. \] (3.8)
Here $w_{ij}$ (for $i,j \in \{r,g,b\}$) are nonnegative weights that measure the contribution of the original channels to each observed channel, and $\star$ denotes two dimensional convolution. We note that at least one of $w_{ri}$, $w_{gi}$ and $w_{bi}$ ($i \in \{r,g,b\}$) is nonzero.

Let $f_i$, $u_i$ and $\eta_i$ (for $i \in \{r,g,b\}$) be the discretized original scene, observed scene and additive noise respectively. Let $H_{ij}$ (for $i,j \in \{r,g,b\}$) be the blurring matrices of appropriate size built according to the discretized point spread functions $h_{ij}$. Then the discretized image formation process can be put into a matrix-vector form:

$$
\begin{align*}
    u_r &= w_{rr} H_{rr} f_r + w_{gr} H_{gr} f_g + w_{br} H_{br} f_b + \eta_r, \\
    u_g &= w_{rg} H_{rg} f_r + w_{gg} H_{gg} f_g + w_{bg} H_{bg} f_g + \eta_g, \\
    u_b &= w_{rb} H_{rb} f_r + w_{gb} H_{gb} f_g + w_{bb} H_{bb} f_b + \eta_b.
\end{align*}
$$

Equation (3.9) can be written as

$$
    u = H(w)f + \eta. 
$$

As in the single channel problem, the use of zero boundary conditions cause the blurring matrices $H_{ij}$ to be Block Toeplitz with Toeplitz Blocks (BTTB). The use of Neumann boundary conditions causes the blurring matrices $H_{ij}$ to be Block Toeplitz plus Hankel with Toeplitz plus Hankel Blocks (BTHTHB).
Chapter 4

RSTLS for high resolution image reconstruction
and color image restoration

4.1 The regularized structured total least squares problem

Traditional methods for solving TLS problems, including the techniques described in chapter 2, are based upon the singular value decomposition. These techniques do not preserve the structure in the blurring matrices in (3.5), (3.7), and (3.10). A new approach, called Structured Total Least Squares (STLS) [59], addresses this problem. The new approach requires that $\tilde{H}$ preserve the structure of $H$. The regularized structured total least squares (RSTLS) problem is often stated as:

$$\min_{\tilde{H}} \|g - \tilde{H}f\|_2^2 + \|\tilde{H} - H\|_F^2 + \alpha \|Rf\|_2^2$$
subject to $\tilde{H}$ has the same structure as $H$.  

As in practice, the RSTLS problem often takes a slightly different form depending on the application. For example, it is often not necessary to use the Frobenius norm to represent the error level in the blurring matrix. Also, for the color image restoration problem, the regularization term consists of multiple regularization parameters. The second term in (4.1) can also be interpreted as a regularization term for the error in $H$. In that interpretation, a regularization parameter for this term is needed. We remark that for the problem of high resolution image reconstruction, it is not necessary to introduce this extra parameter. However, for the problem of color image restoration, we found it necessary to do so.

Recall that an important issue in the regularization is the choice of the regularization parameter(s). For the problem of high resolution image reconstruction, in which only one parameter needs to be determined. The L-curve method can be used to find its optimal value. For the problem of color image restoration, more than one parameter need to be determined. The procedure for finding the optimal values in this problem will be explained later. In the sequel, our focus is on solving (4.1) or a modified form of it for fixed regularization parameter(s).

The main contribution of this chapter includes developing RSTLS algorithms for high resolution image reconstruction and color image restoration, and proposing efficient preconditioners for the linear systems encountered in our RSTLS algorithms.
4.2 RSTLS algorithm for single channel image deblurring

Pruessner and O’Leary [56] extended the Rosen et al. [59] technique to include regularization and demonstrated its use on image deblurring. They considered 1-, 2- and $\infty$-norm minimization for one and two dimensional problems. For the sake of self-containment, we present their 2-norm minimization algorithm for the one dimensional problem first.

4.2.1 The one dimensional problem

For the one dimensional case, the blurring matrix $H$ is an $n \times n$ Toeplitz matrix of bandwidth $m$ when zero boundary conditions are used. Therefore the RSTLS method requires that $\tilde{H}$ be an $n \times n$ Toeplitz matrix of bandwidth $m$, and so is the error matrix $E = \tilde{H} - H$, therefore, $E$ has the form

$$E = \text{Toeplitz}([z_0, z_1, \ldots, z_m, 0, \ldots, 0]^T, [z_0, z_{-1}, \ldots, z_{-m}, 0, \ldots, 0]^T).$$

Then the RSTLS problem can be stated as:

$$\min_{E, f} \|g - (H + E)f\|_2^2 + \|E\|_F^2 + \alpha \|Rf\|_2^2$$
subject to $E$ having the form of (4.2).

Let $z$ be the vector of length $2m + 1$ that contains the distinct elements in $E$,

$$z = [z_{-m} \ldots z_{-1} z_0 z_1 \ldots z_m]^T.$$  \hspace{1cm} (4.4)

Define $D$ to be the following $(2m + 1) \times (2m + 1)$ diagonal matrix:

$$D = \text{diag}(\sqrt{n-m}, \sqrt{n-m+1}, \ldots \sqrt{n}, \ldots \sqrt{n-m+1}, \sqrt{n-m}).$$

then it is clear that $\|E\|_F = \|Dz\|_2$. Let $r(z, f) = g - (H + E)f$, then (4.3) can be written as:

$$\min_{z, f} \|r(z, f)\|_2^2 + \|Dz\|_2^2 + Rf\|_2^2.$$  \hspace{1cm} (4.5)

An iterative algorithm for solving (4.5) will now be described. Given some current guess $(z, f)$ of the solution to (4.5), the idea is to find $\Delta z$ and $\Delta f$ that approximately solves the following minimization problem at each iteration:

$$\min_{\Delta z, \Delta f} \|r(z + \Delta z, f + \Delta f)\|_2^2 + \|D(z + \Delta z)\|_2^2 + \alpha \|R(f + \Delta f)\|_2^2.$$  \hspace{1cm} (4.6)
The minimization required by (4.6) can be done by a linear approximation to 
\( r(z + \Delta z, f + \Delta f) \). Let \( \Delta E \) represent a structured change in \( E \) that corresponds to a change of \( \Delta z \) in \( z \), since

\[
\begin{align*}
    r(z + \Delta z, f + \Delta f) &= g - (H + E + \Delta E)(f + \Delta f) \\
    &= g - (H + E)f - (\Delta E)f - (H + E)\Delta f - \Delta E\Delta f \\
    &= r(z, f) - (\Delta E)f - (H + E)\Delta f - \Delta E\Delta f
\end{align*}
\]

by neglecting the second-order term \( \Delta E\Delta f \), we get

\[
    r(z + \Delta z, f + \Delta f) \approx r(z, f) - (\Delta E)f - (H + E)\Delta f. \quad (4.7)
\]

Next we try to find a matrix \( X \) such that:

\[
    Xz = Ef. \quad (4.8)
\]

It is easy to verify that the following \( n \times (2m + 1) \) Toeplitz matrix satisfies the above equation:

\[
    X = \text{Toeplitz}([f_{m+1}, f_{m+2}, \ldots, f_n, 0, \ldots, 0]^T, [f_{m+1}, f_m, \ldots, f_1, 0, \ldots, 0]^T). \quad (4.9)
\]

Therefore,

\[
    X\Delta z = (\Delta E)f.
\]

Now (4.7) becomes

\[
    r(z + \Delta z, f + \Delta f) \approx r(z, f) - X\Delta z - (H + E)\Delta f. \quad (4.10)
\]

The minimization problem (4.6), with \( r(z + \Delta z, f + \Delta f) \) replaced by (4.10), becomes

\[
    \min_{\Delta z, \Delta f} \left\| \begin{bmatrix} X & H + E \\ D & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta f \end{bmatrix} + \begin{bmatrix} -r \\ D\Delta z \end{bmatrix} \right\|_2^2. \quad (4.11)
\]

Rosen et al. [59] showed that (4.11) is basically a Gauss-Newton step. To start the iterative algorithm, we can use \( E = 0 \) and the least square solution \( f = f_{ls} \) as an initial guess.

Note that (4.11) is a simple least squares problem, and can be solved by either a direct method such as QR factorization or a Krylov subspace method such as CGLS [10] or LSQR [55]. In §4.5, we will discuss how to solve this system efficiently.
The regularized structured total least squares algorithm for fixed regularization parameter $\alpha$ is summarized in Algorithm 4.1.

**Algorithm 4.1**

1. Set $E = 0$, $z = 0$, compute the least squares solution $f_{ls}$, set $f = f_{ls}$, set $r = g - Hf$, construct $X$ according to (4.9).

2. Repeat
   
   (a) $\min_{\Delta z, \Delta f} \left\| \begin{bmatrix} X & H + E \\ D & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta f \end{bmatrix} + \begin{bmatrix} -r \\ Dz \end{bmatrix} \right\|_2^2$
   
   (b) Set $f = f + \Delta f$, $z = z + \Delta z$.
   
   (c) Construct $E$ from $z$, construct $X$ from $f$ according to (4.9), compute $r = g - (H + E)f$.

until $\|\Delta f\|, \|\Delta z\| \leq \epsilon$.

For the one dimensional problem with Neumann boundary condition, the blurring matrix $H$ is in the form of (3.5), which is an $n \times n$ Toeplitz plus Hankel matrix. Therefore, $\tilde{H}$ and the error matrix $E = \tilde{H} - H$ must have the same form. Let

$$E = E_l J + E_m J + E_r J$$

(4.12)

with

$$E_l = \text{Toeplitz}([0, \ldots, 0]^T, [0, \ldots, 0, z_m, z_{m-1}, \ldots, z_1]^T),$$

$$E_m = \text{Toeplitz}([z_0, z_1, \ldots, z_m, 0, \ldots, 0]^T, [z_0, z_{-1}, \ldots, z_{-m}, 0, \ldots, 0]^T),$$

$$E_r = \text{Toeplitz}([0, \ldots, 0, z_{-m}, z_{-m+1}, \ldots, z_{-1}]^T, [0, \ldots, 0]^T).$$

Define the $2m+1$ vector $z$ the same as (4.4). Let $D = \sqrt{n}I$, then the RSTLS problem can be stated as

$$\min_{z,f} \|g - (H + E)f\|_2^2 + \|Dz\|_2^2 + \alpha \|Rf\|_2^2.$$  

(4.13)

For easy computation, here we use $\|Dz\|_2^2$, which is equal to $\|E_l\|_F^2 + \|E_m\|_F^2 + \|E_r\|_F^2$, instead of $\|E\|_F^2$ to represent the error level in the blurring matrix.

Define the $n \times (2m + 1)$ matrix $X$ as the following:

$$X = \text{Toeplitz}(f_{m+1}, f_{m+2}, \ldots, f_n, 0, \ldots, 0)^T, [f_{m+1}, f_m, \ldots, f_1, 0, \ldots, 0]^T) + \text{Toeplitz}(0, \ldots, 0, f_n, f_{n-1}, \ldots, f_{n-m+1})^T, [0, \ldots, 0]^T) + \text{Toeplitz}(0, \ldots, 0)^T, [0, \ldots, 0, f_1, f_2, \ldots, f_m]^T).$$

Then $X$ satisfies $Xz = Ef$, therefore Algorithm 4.1 can be used for Toeplitz plus Hankel matrices, except that now we construct $E$ according to (4.12) and $X$ according to (4.14).
4.2.2 The two dimensional problem with separable blurring function

When zero boundary conditions are used, the blurring matrix is BTTB for two dimensional problems. In the case that the blurring function is separable, the blurring matrix is further structured to be the Kronecker product of two Toeplitz matrices. That is, the blurring matrix has the form of \( H = H_1 \otimes H_2 \), where \( H_1 \) is an \( n_1 \times n_1 \) Toeplitz matrix of bandwidth \( m_1 \) and \( H_2 \) is an \( n_2 \times n_2 \) Toeplitz matrix of bandwidth \( m_2 \). Therefore, the RSTLS solution requires that \( \tilde{H} \) be in the form of \( \tilde{H}_1 \otimes \tilde{H}_2 \), where \( \tilde{H}_1 \) has the same structure as \( H_1 \) and \( \tilde{H}_2 \) has the same structure as \( H_2 \). We single out this case because previous work does not distinguish it from the inseparable cases, and solving this case serves as an important step towards solving the RSTLS problem for high resolution image reconstruction.

For two dimensional problems with separable blurring functions, the resulting RSTLS problem can be stated as:

\[
\begin{align*}
\min \| g - (\tilde{H}_1 \otimes \tilde{H}_2)f \|^2_2 + \| \tilde{H}_1 - H_1 \|^2_F + \| \tilde{H}_2 - H_2 \|^2_F + \alpha \| Rf \|^2_2 \\
\text{subject to } \tilde{H}_1, \tilde{H}_2 \text{ have the same structure as } H_1, H_2.
\end{align*}
\]

(4.14)

Here we use \( \| \tilde{H}_1 - H_1 \|^2_F + \| \tilde{H}_2 - H_2 \|^2_F \) instead of \( \| (\tilde{H}_1 \otimes \tilde{H}_2) - (H_1 \otimes H_2) \|^2_F \) to represent the error level in the blurring matrix because it leads to simpler equations.

Let \( E_1 = \tilde{H}_1 - H_1 \), and \( E_2 = \tilde{H}_2 - H_2 \), then \( E_1 \) is an \( n_1 \times n_1 \) Toeplitz matrix of bandwidth \( m_1 \) and \( E_2 \) is an \( n_2 \times n_2 \) Toeplitz matrix of bandwidth \( m_2 \). Let

\[
\begin{align*}
E_1 &= Toeplitz([z_0, z_1, \ldots, z_{m_1}, 0, \ldots, 0]^T, [z_0, z_{-1}, \ldots, z_{-m_1}, 0, \ldots, 0]^T), \\
E_2 &= Toeplitz([v_0, v_1, \ldots, v_{m_2}, 0, \ldots, 0]^T, [v_0, v_{-1}, \ldots, v_{-m_2}, 0, \ldots, 0]^T).
\end{align*}
\]

(4.15) \hspace{1cm} (4.16)

Let \( z \) and \( v \) be the vectors of length \( 2m_1 + 1 \) and \( 2m_2 + 1 \) that contain the distinct elements in \( E_1 \) and \( E_2 \) respectively, that is,

\[
\begin{align*}
z &= \left[ \begin{array}{cccc}
z_{-m_1} & \ldots & z_0 & \ldots & z_{m_1} \end{array} \right]^T, \\
v &= \left[ \begin{array}{cccc}
v_{-m_2} & \ldots & v_0 & \ldots & v_{m_2} \end{array} \right]^T.
\end{align*}
\]

(4.17)

Define \( D_1 \) and \( D_2 \) as

\[
D_1 = diag(\sqrt{n_1 - m_1}, \sqrt{n_1 - m_1 + 1}, \ldots, \sqrt{n_1}, \ldots, \sqrt{n_1 - m_1 + 1}, \sqrt{n_1 - m_1}).
\]


\[ D_2 = \text{diag}(\sqrt{n_2-m_2}, \sqrt{n_2-m_2+1}, \ldots \sqrt{n_2}, \ldots \sqrt{n_2-m_2+1}, \sqrt{n_2-m_2}) , \]

then the RSTLS problem becomes:

\[
\begin{aligned}
\min_{z,v,f} \|g - ((H_1 + E_1) \otimes (H_2 + E_2))f\|_2^2 + \|D_1z\|_2^2 + \|D_2v\|_2^2 + \alpha \|Rf\|_2^2. \\
\text{(4.18)}
\end{aligned}
\]

Let \( r(z,v,f) = g - ((H_1 + E_1) \otimes (H_2 + E_2))f \). Similar to what was done for the one dimensional case, we can use a linear approximation for \( r(z + \Delta z, v + \Delta v, f + \Delta f) \):

\[
\begin{aligned}
r(z + \Delta z, v + \Delta v, f + \Delta f) &= g - ((H_1 + E_1 + \Delta E_1) \otimes (H_2 + E_2 + \Delta E_2))(f + \Delta f) \\
&= g - ((H_1 + E_1) \otimes (H_2 + E_2))f - ((H_1 + E_1) \otimes (H_2 + E_2))\Delta f \\
&\quad - (\Delta E_1 \otimes (H_2 + E_2))f - ((H_1 + E_1) \otimes \Delta E_2)f \\
&\quad - (\Delta E_1 \otimes \Delta E_2)f - (\Delta E_1 \otimes (H_2 + E_2))\Delta f \\
&\quad - ((H_1 + E_1) \otimes \Delta E_2)\Delta f - (\Delta E_1 \otimes \Delta E_2)\Delta f.
\end{aligned}
\] (4.19)

By ignoring higher order terms, we get

\[
\begin{aligned}
r(z + \Delta z, v + \Delta v, f + \Delta f) &\approx r(z,v,f) - ((H_1 + E_1) \otimes (H_2 + E_2))\Delta f \\
&\quad - (\Delta E_1 \otimes (H_2 + E_2))f - ((H_1 + E_1) \otimes \Delta E_2)f.
\end{aligned}
\] (4.20)

Next we try to find matrices \( X \) and \( Y \) such that

\[
\begin{aligned}
X\Delta z &= (\Delta E_1 \otimes (H_2 + E_2))f, \\
Y\Delta v &= ((H_1 + E_1) \otimes \Delta E_2)f.
\end{aligned}
\] (4.21, 4.22)

Partition \( f \) as the following:

\[
f = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n_1} \end{bmatrix},
\] (4.23)
where \( f_1, f_2, \ldots, f_{n_1} \) are vectors of length \( n_2 \). Denote the \( j \)th element of \( f_i \) as \( f_{i,j} \), then the \( n_1n_2 \times (2m_1 + 1) \) matrix \( X \) defined as the following, satisfies (4.21).

\[
X = \begin{bmatrix}
(H_2 + E_2)f_{m_1+1} & \cdots & (H_2 + E_2)f_{1} & 0 \\
\vdots & & \ddots & \ddots \\
(H_2 + E_2)f_{n_1-m_1} & \cdots & (H_2 + E_2)f_{1} & \vdots \\
\vdots & & \ddots & \ddots & \ddots \\
(H_2 + E_2)f_{n_1} & \cdots & (H_2 + E_2)f_{m_1+1} & 0 \\
\end{bmatrix}.
\]  

(4.24)

Let

\[
[y_1 \ y_2 \ \cdots \ y_{n_2}] = (H_1 + E_1) \begin{bmatrix} f_1 & f_2 & \cdots & f_{n_1} \end{bmatrix}^T.
\]

Denote the \( j \)th element of \( y_i \) by \( y_{i,j} \), let \( Y_i \) be the \( n_2 \times (2m_2 + 1) \) matrix defined as

\[
Y_i = Toeplitz([y_{m_2+1,i}, y_{m_2+2,i}, \ldots, y_{n_2,i}, 0, \ldots, 0]^T),
\]

for \( i = 1, 2, \ldots, n_1 \), then the matrix \( Y \) defined as the following satisfies (4.22):

\[
Y = \begin{bmatrix}
Y_1 \\
Y_2 \\
\vdots \\
Y_{n_1}
\end{bmatrix}.
\]  

(4.25)

Now (4.20) becomes

\[
\begin{aligned}
& r(z + \Delta z, v + \Delta v, f + \Delta f) \\
\approx & \quad r(z, v, f) - ((H_1 + E_1) \otimes (H_2 + E_2))\Delta f - X\Delta z - Y\Delta v.
\end{aligned}
\]  

(4.26)

Therefore, given some current guess of the solution \((z, v, f)\), the minimization problem (4.18) can be approximated by solving

\[
\min_{\Delta z, \Delta v, \Delta f} \left\| \begin{bmatrix} X & Y & (H_1 + E_1) \otimes (H_2 + E_2) \\
D_1 & 0 & 0 \\
0 & D_2 & 0 \\
0 & 0 & \sqrt{\alpha R} \end{bmatrix} \begin{bmatrix} \Delta z \\
\Delta v \\
\Delta f \end{bmatrix} + \begin{bmatrix} -r \\
D_1z \\
D_2v \\
\sqrt{\alpha R}f \end{bmatrix} \right\|_2^2.
\]  

(4.27)
Again this is basically a Gauss-Newton step. The RSTLS algorithm for BTTB matrices in the form of $H_1 \otimes H_2$ is summarized below:

**Algorithm 4.2**

1. Set $E_1 = 0$, $E_2 = 0$, $z = 0$, $v = 0$.
   - Compute the least squares solution $f_0$.
   - Set $f = f_0$, set $r = g - ((H_1 + E_1) \otimes (H_2 + E_2))f$, construct $X$ and $Y$ according to (4.24) and (4.25).

2. Repeat
   - (a) Solve (4.27) for $\Delta f$, $\Delta z$, $\Delta v$.
   - (b) Set $f = f + \Delta f$, $z = z + \Delta z$, $v = v + \Delta v$.
   - (c) Construct $E_1$ from $z$, $E_2$ from $v$.
   - Construct $X$ and $Y$ from $f$ according to (4.24) and (4.25), compute $r = g - ((H_1 + E_1) \otimes (H_2 + E_2))f$.
   - Until $\|\Delta f\|_2, \|\Delta z\|_2, \|\Delta v\|_2 \leq \epsilon$.

When Neumann boundary conditions are used, the blurring matrix $H$ is BTHTHB for two dimensional problems. That is, $H$ has a block level structure as (3.5) and each block also has the form of (3.5). In the case that the blurring function is separable, the blurring matrix is further structured to be the Kronecker product of two Toeplitz plus Hankel matrices. That is, the blurring matrix has the form of $H = \tilde{H}$, where $\tilde{H}$ is an $n_1 \times n_1$ Toeplitz plus Hankel matrix of bandwidth $m_1$, and $H_2$ is an $n_2 \times n_2$ Toeplitz plus Hankel matrix of bandwidth $m_2$. Therefore, the STLS solution would require that $\tilde{H}$ be in the form of $\tilde{H}_1 \otimes \tilde{H}_2$, where $\tilde{H}_1$ and $\tilde{H}_2$ are simple Toeplitz plus Hankel matrices.

Define $E_{1\ell}$, $E_{1m}$, and $E_{1r}$ as the following:

\[
E_{1\ell} = \text{Toeplitz}([0, \ldots, 0]^T, [0, \ldots, 0, z_m, z_{m-1}, \ldots, z_1]^T),
\]

\[
E_{1m} = \text{Toeplitz}([z_0, z_1, \ldots, z_m, 0, \ldots, 0]^T, [z_0, z_1, \ldots, z_{m-1}, 0, \ldots, 0]^T),
\]

\[
E_{1r} = \text{Toeplitz}([0, \ldots, 0, z_m, z_{m+1}, \ldots, z_1]^T, [0, \ldots, 0]^T).
\]

Define $E_{2\ell}$, $E_{2m}$, and $E_{2r}$ similarly. Let $E_1 = \tilde{H}_1 - H_1$ and $E_2 = \tilde{H}_2 - H_2$, then $E_1$ and $E_2$ have the form of

\[
E_1 = E_{1\ell}J + E_{1m} + E_{1r}J,
\]

\[
E_2 = E_{2\ell}J + E_{2m} + E_{2r}J.
\]

Let $z$ and $v$ be the vectors that contain the distinct elements in $E_1$ and $E_2$ respectively, then $z$ and $v$ will have the same form as (4.17). Let $D_1 = \sqrt{m_1}I$ and $D_2 = \sqrt{m_2}I$, then the RSTLS problem can be stated as

\[
\min_{z,v,f} \|g - ((H_1 + E_1) \otimes (H_2 + E_2))f\|_2^2 + \|D_1 z\|_2^2 + \|D_2 v\|_2^2 + \alpha \|Rf\|_2^2. \quad (4.28)
\]
For easy computation, here we use \( \|D_1 z\|_F^2 + \|D_2 \mathbf{v}\|_F^2 \), which is equal to \( \|E_{1r}\|_F^2 + \|E_{1r}\|_F^2 + \|E_{2r}\|_F^2 + \|E_{2r}\|_F^2 \), instead of \( \|E_1\|_F^2 + \|E_2\|_F^2 \) to represent the error level in the blurring matrix.

If we partition \( \mathbf{f} \) the same way as in (4.23), then the \((n_1 n_2) \times (2m_1 + 1)\) matrix \( \mathbf{X} \) defined as

\[
\mathbf{X} = \begin{bmatrix}
(H_2 + E_2) f_{m_1+1} & \cdots & (H_2 + E_2) f_1 & 0 \\
\vdots & \ddots & \vdots & \vdots \\
(H_2 + E_2) f_{n_1-m_1} & \cdots & (H_2 + E_2) f_1 \\
\vdots & \ddots & \vdots & \vdots \\
(H_2 + E_2) f_{n_1} & \cdots & (H_2 + E_2) f_{m_1+1} \\
0 & \cdots & (H_2 + E_2) f_{n_1} & \cdots & (H_2 + E_2) f_{n_1-m_1}
\end{bmatrix}
\]

\[
= \begin{bmatrix}
0 & \cdots & 0 \\
(H_2 + E_2) f_{n_1} & \cdots & 0 \\
\vdots & \ddots & \vdots \\
(H_2 + E_2) f_{n_1-m_1+2} & \cdots & (H_2 + E_2) f_{n_1-m_1+2} & \cdots & 0 \\
(H_2 + E_2) f_{n_1-m_1+1} & (H_2 + E_2) f_{n_1-m_1+2} & \cdots & (H_2 + E_2) f_{n_1} & 0 \\
0 & (H_2 + E_2) f_1 & \cdots & (H_2 + E_2) f_{m_1-1} & (H_2 + E_2) f_{m_1} \\
0 & (H_2 + E_2) f_1 & \cdots & (H_2 + E_2) f_{m_1-1} & \cdots & (H_2 + E_2) f_1 \\
0 & \cdots & \cdots & \cdots & \cdots & 0 \\
\end{bmatrix} 
+ \begin{bmatrix}
0 & \cdots & 0 \\
0 & (H_2 + E_2) f_1 & \cdots & (H_2 + E_2) f_{m_1-1} & (H_2 + E_2) f_{m_1} \\
\vdots & \cdots & \cdots & \cdots & \cdots & (H_2 + E_2) f_1 \\
\vdots & \cdots & \cdots & \cdots & \cdots & 0 \\
\end{bmatrix}
\]

satisfies (4.21).

Let

\[
\begin{bmatrix}
\mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_{n_2}
\end{bmatrix} = (H_1 + E_1) \begin{bmatrix}
\mathbf{f}_1 & \mathbf{f}_2 & \cdots & \mathbf{f}_{n_1}
\end{bmatrix}^T.
\]

Denote the \( j \)th element of \( \mathbf{y}_i \) by \( y_{i,j} \). Let \( Y_i \) be the \( n_2 \times (2m_2 + 1) \) matrix defined as

\[
Y_i = \text{Toeplitz}\left( \begin{bmatrix} y_{m_2+1,i} & y_{m_2+2,i} & \cdots & y_{n_2,i} & 0, \ldots, 0 \end{bmatrix}^T, \\
[y_{m_2+1,i}, y_{m_2,i}, \ldots, y_{1,i}, 0, \ldots, 0]^T \right) \\
+ \text{Toeplitz}\left( \begin{bmatrix} 0, \ldots, 0, y_{n_2,i}, y_{n_2-1,i}, \ldots, y_{2-m_2+1,i} \end{bmatrix}^T, \begin{bmatrix} 0, \ldots, 0 \end{bmatrix}^T \right) \\
+ \text{Toeplitz}\left( \begin{bmatrix} 0, \ldots, 0 \end{bmatrix}^T, \begin{bmatrix} 0, \ldots, 0, y_{1,i}, y_{2,i}, \ldots, y_{m_2,i} \end{bmatrix}^T \right).
\]
Then the matrix $Y$ defined as

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}$$

satisfies (4.22).

Therefore, Algorithm 4.2 can be used for separable BTHTHB matrices, except that now we construct $X$ and $Y$ according to (4.29) and (4.30).

### 4.3 RSTLS algorithm for high resolution image reconstruction

For this problem, the error in the displacement estimate is the only cause of the error in the blurring matrix $H$. Therefore, $\tilde{H}$ has to be in the following form

$$\tilde{H} = \sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1\ell_2} \tilde{H}_{\ell_1\ell_2}^x \otimes \tilde{H}_{\ell_1\ell_2}^y,$$

where $\tilde{H}_{\ell_1\ell_2}^x$ and $\tilde{H}_{\ell_1\ell_2}^y$ have the form of (3.6).

Let $E_{\ell_1\ell_2}^x = \tilde{H}_{\ell_1\ell_2}^x - H_{\ell_1\ell_2}^x$. Since both $\tilde{H}_{\ell_1\ell_2}^x$ and $H_{\ell_1\ell_2}^x$ have the form of (3.6), $E_{\ell_1\ell_2}^x$ must have the form of

$$E_{\ell_1\ell_2}^x = \begin{cases} E_{\ell_1\ell_2}^x \\ E_{\ell_1\ell_2}^x J + E_{\ell_1\ell_2}^x \zeta + E_{\ell_1\ell_2}^x \zeta \end{cases} \quad \text{under Neumann boundary conditions}$$

Here

$$E_{\ell_1\ell_2}^x = \frac{\delta_{\ell_1\ell_2}}{L} \cdot \text{Toeplitz}([0, \ldots, 0, 1, 0, \ldots, 0]^T, [0, \ldots, 0, -1, 0, \ldots, 0]^T),$$

$$E_{\ell_1\ell_2}^x = \frac{\delta_{\ell_1\ell_2}}{L} \cdot \text{Toeplitz}([0, \ldots, 0]^T, [0, \ldots, 0, 1, 0, \ldots, 0]^T),$$

$$E_{\ell_1\ell_2}^x = \frac{\delta_{\ell_1\ell_2}}{L} \cdot \text{Toeplitz}([0, \ldots, 0, -1, 0, \ldots, 0]^T, [0, \ldots, 0]^T).$$

$E_{\ell_1\ell_2}^y$ is defined similarly.
Let \( z \) be the vector that contains the displacement errors of each sensor in the \( x \) direction, and \( v \) be the vector that contains the displacement errors of each sensor in the \( y \) direction, that is

\[
\begin{align*}
    z &= \begin{bmatrix} \delta \epsilon_{00}^x & \delta \epsilon_{01}^x & \cdots & \delta \epsilon_{L-1,L-2}^x \end{bmatrix}^T, \\
    v &= \begin{bmatrix} \delta \epsilon_{00}^y & \delta \epsilon_{01}^y & \cdots & \delta \epsilon_{L-1,L-2}^y \end{bmatrix}^T.
\end{align*}
\]

Then the RSTLS problem can be stated as

\[
\min_{z, v, f} \| r(z, v, f) \|^2_2 + \| z \|^2_2 + \| v \|^2_2 + \alpha \| Rf \|^2_2.
\]

Here

\[
r(z, v, f) = g - \left( \sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1 \ell_2} \left( H_{\ell_1 \ell_2}^x + E_{\ell_1 \ell_2}^y \right) \otimes \left( H_{\ell_1 \ell_2}^y + E_{\ell_1 \ell_2}^y \right) \right) f.
\]

As was done in the previous section, a linearization of \( r(z + \Delta z, v + \Delta v, f + \Delta f) \) yields

\[
 r(z + \Delta z, v + \Delta v, f + \Delta f) \approx r(z, v, f)
\]

\[
- \left( \sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1 \ell_2} \left( H_{\ell_1 \ell_2}^x + E_{\ell_1 \ell_2}^y \right) \otimes \left( H_{\ell_1 \ell_2}^y + E_{\ell_1 \ell_2}^y \right) \right) \Delta f
\]

\[
- \left( \sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1 \ell_2} \left( \Delta E_{\ell_1 \ell_2}^x \right) \otimes \left( H_{\ell_1 \ell_2}^y + E_{\ell_1 \ell_2}^y \right) \right) f
\]

\[
- \left( \sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1 \ell_2} \left( H_{\ell_1 \ell_2}^x + E_{\ell_1 \ell_2}^y \right) \otimes \Delta E_{\ell_1 \ell_2}^y \right) f.
\]

Next we need matrices \( X \) and \( Y \) such that

\[
Xz = \left( \sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1 \ell_2} \left( E_{\ell_1 \ell_2}^x \otimes \left( H_{\ell_1 \ell_2}^y + E_{\ell_1 \ell_2}^y \right) \right) \right) f,
\]

\[
Yv = \left( \sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1 \ell_2} \left( \left( H_{\ell_1 \ell_2}^x + E_{\ell_1 \ell_2}^x \right) \otimes E_{\ell_1 \ell_2}^y \right) \right) f.
\]
Partition $f$ as the following:

$$
f = \begin{bmatrix}
    f_1 \\
    f_2 \\
    \vdots \\
    f_{M_1}
\end{bmatrix},
$$

where $f_1, f_2, \ldots, f_{M_1}$ are vectors of length $M_2$. Let the $(M_1 M_2) \times 1$ matrix $F^{\ell_1 \ell_2}$ be defined as

$$
F^{\ell_1 \ell_2} = - \begin{bmatrix}
    (H^{y}_{\ell_1 \ell_2} + E^{y}_{\ell_1 \ell_2}) f_{L/2+1} \\
    \vdots \\
    (H^{y}_{\ell_1 \ell_2} + E^{y}_{\ell_1 \ell_2}) f_{M_1} \\
    0
\end{bmatrix} + \begin{bmatrix}
    0 \\
    (H^{y}_{\ell_1 \ell_2} + E^{y}_{\ell_1 \ell_2}) f_1 \\
    \vdots \\
    (H^{y}_{\ell_1 \ell_2} + E^{y}_{\ell_1 \ell_2}) f_{M_1-L/2}
\end{bmatrix}.
$$

Let the $(M_1 M_2) \times 1$ matrix $X^{\ell_1 \ell_2}$ be defined as

$$
X^{\ell_1 \ell_2} = D_{\ell_1 \ell_2} F^{\ell_1 \ell_2}
$$

when zero boundary conditions are used, and

$$
X^{\ell_1 \ell_2} = D_{\ell_1 \ell_2} \left( F^{\ell_1 \ell_2} - \begin{bmatrix}
    0 \\
    (H^{y}_{\ell_1 \ell_2} + E^{y}_{\ell_1 \ell_2}) f_{M_1} \\
    \vdots \\
    (H^{y}_{\ell_1 \ell_2} + E^{y}_{\ell_1 \ell_2}) f_{M_1-L/2+1}
\end{bmatrix} + \begin{bmatrix}
    (H^{y}_{\ell_1 \ell_2} + E^{y}_{\ell_1 \ell_2}) f_{L/2} \\
    \vdots \\
    (H^{y}_{\ell_1 \ell_2} + E^{y}_{\ell_1 \ell_2}) f_{1}
\end{bmatrix} \right)
$$

when Neumann boundary conditions are used. It follows that the matrix $X$ defined as

$$
X = \frac{1}{L} \begin{bmatrix}
    X^{00} & X^{01} & \ldots & X^{L-1,L-2} & X^{L-1,L-1}
\end{bmatrix}
$$

satisfies (4.32).

Let

$$
\begin{bmatrix}
    y_1^{\ell_1 \ell_2} \\
    y_2^{\ell_1 \ell_2} \\
    \vdots \\
    y_{M_2}^{\ell_1 \ell_2}
\end{bmatrix} = (H^{x}_{\ell_1 \ell_2} + E^{x}_{\ell_1 \ell_2}) \begin{bmatrix}
    f_1 \\
    f_2 \\
    \vdots \\
    f_{M_1}
\end{bmatrix}^T.
$$
Denote the $j$th element of $y_i^{\ell_1 \ell_2}$ as $y_{i,j}^{\ell_1 \ell_2}$. Let $F_i^{\ell_1 \ell_2}$ be the matrix of size $M_2 \times 1$ defined as

$$F_i^{\ell_1 \ell_2} = -\begin{bmatrix} y_{L/2+1,i}^{\ell_1 \ell_2} \\ \vdots \\ y_{M_1,i}^{\ell_1 \ell_2} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ y_{1,i}^{\ell_1 \ell_2} \\ \vdots \\ y_{M_2-L/2,i}^{\ell_1 \ell_2} \end{bmatrix}.$$ 

Let $Y_i^{\ell_1 \ell_2} = F_i^{\ell_1 \ell_2}$ when zero boundary conditions are used, and

$$Y_i^{\ell_1 \ell_2} = F_i^{\ell_1 \ell_2} - \begin{bmatrix} 0 \\ y_{L/2,i}^{\ell_1 \ell_2} \\ \vdots \\ y_{M_2-L/2+1,i}^{\ell_1 \ell_2} \end{bmatrix} + \begin{bmatrix} y_{L/2,i}^{\ell_1 \ell_2} \\ \vdots \\ y_{1,i}^{\ell_1 \ell_2} \\ 0 \end{bmatrix}$$

when Neumann boundary conditions are used. Let

$$Y^{\ell_1 \ell_2} = D_{\ell_1 \ell_2} \begin{bmatrix} Y_1^{\ell_1 \ell_2} \\ Y_2^{\ell_1 \ell_2} \\ \vdots \\ Y_{M_2}^{\ell_1 \ell_2} \end{bmatrix}.$$ 

Then the matrix $Y$ defined as

$$Y = \frac{1}{L} \begin{bmatrix} Y^{00} & Y^{01} & \ldots & Y^{L-1,L-2} & Y^{L-1,L-1} \end{bmatrix}$$

satisfies (4.33).

Now (4.31) becomes

$$r(z + \Delta z, v + \Delta v, f + \Delta f) \approx r(z, v, f) - \left( \sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1 \ell_2} (H_{\ell_1 \ell_2}^x + E_{\ell_1 \ell_2}^x) \otimes (H_{\ell_1 \ell_2}^y + E_{\ell_1 \ell_2}^y) \right) \Delta f - X \Delta z - Y \Delta v. \quad (4.36)$$

Therefore, given some current guess of the solution $(z, v, f)$, the minimization problem, with $r(z + \Delta z, v + \Delta v, f + \Delta f)$ replaced by (4.36), becomes

$$\min_{\Delta z, \Delta v, \Delta f} \left\| \begin{bmatrix} 0 & X & Y \\ I & 0 & 0 \\ 0 & I & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta v \\ \Delta f \end{bmatrix} + \begin{bmatrix} -r \\ z \\ v \end{bmatrix} \right\|_2^2. \quad (4.37)$$
Here $\tilde{H}$ is the blurring matrix that corresponds to the current displacement estimates, or
\[
\tilde{H} = \sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1\ell_2} (H_{x,\ell_1\ell_2}^x + E_{x,\ell_1\ell_2}^x) \otimes (H_{y,\ell_1\ell_2}^y + E_{y,\ell_1\ell_2}^y).
\] (4.38)

The RSTLS algorithm for high-resolution image reconstruction is summarized below.

<table>
<thead>
<tr>
<th>Algorithm 4.3</th>
</tr>
</thead>
</table>
| 1. Set $E_{x,\ell_1\ell_2}^x = 0$, $E_{y,\ell_1\ell_2}^y = 0$, $z = 0$, $v = 0$,
| compute the least squares solution $f_{ls}$.
| set $f = f_{ls}$, set $r = g - (\sum_{\ell_1=0}^{L-1} \sum_{\ell_2=0}^{L-1} D_{\ell_1\ell_2} (H_{x,\ell_1\ell_2}^x \otimes H_{y,\ell_1\ell_2}^y))f$,
| construct $X$ and $Y$ according to (4.34) and (4.35).
| 2. Repeat
| (a) Solve (4.37) for $\Delta f$, $\Delta z$, $\Delta v$.
| (b) Set $f = f + \Delta f$, $z = z + \Delta z$, $v = v + \Delta v$.
| (c) Construct $E_{x,\ell_1\ell_2}^x$ from $z$, $E_{y,\ell_1\ell_2}^y$ from $v$,
| construct $X$ and $Y$ from $f$ according to (4.34) and (4.35),
| construct $\tilde{H}$ according to (4.38),
| compute $r = g - \tilde{H}f$.
| until $\|\Delta f\|$, $\|\Delta z\|$, $\|\Delta v\| \leq \epsilon$. |

Compared to the algorithm proposed by Ng et al. in [48], Algorithm 4.3 solves the exact regularization of the STLS problem. The Ng et al. algorithm did not use the exact constraint (eq. (3.7) in [48]), instead it used a constraint (Eq. (3.9) in [48]) that approximates the exact one. Had the Ng et al. algorithm used the exact constraint, it would be solving exactly the same problem as Algorithm 4.3. Algorithm 4.3 is also faster. The reader shall see later that this algorithm needs no more than 6 iterations to converge for a fixed regularization parameter, while the algorithm in [48] needs dozens of iterations to converge (see Figure 2 on page 40 of [48]). For both algorithms, the major cost of one iteration is that of solving a linear least squares problem of similar size.

4.4 RSTLS for color image restoration

4.4.1 Regularizing a color image

Regularizing a color image is quite different from regularizing a single channel image. One widely used regularization technique for color image restoration is that proposed by Galatsanos et al. in [28].

An important concept in the Galatsanos et al. regularization technique is the “weighted 3D Laplacian”. Let
\[
\beta = [\beta_{rg} \quad \beta_{rb} \quad \beta_{gb}]^T = \left[ \frac{\|f_r\|_2}{\|f_g\|_2} \quad \frac{\|f_r\|_2}{\|f_b\|_2} \quad \frac{\|f_g\|_2}{\|f_b\|_2} \right]^T,
\] (4.39)
and let $L$ be the 2D Laplacian matrix of appropriate size. The “weighted 3D Laplacian” matrix is defined as follows:

$$ R(\beta) = \begin{bmatrix} 2I + L & -\beta_{rg}I & -\beta_{rb}I \\ -\frac{1}{\beta_{rg}}I & 2I + L & -\beta_{gb}I \\ -\frac{1}{\beta_{rb}}I & -\frac{1}{\beta_{gb}}I & 2I + L \end{bmatrix}. $$

Note that

$$ R(\beta) = D\hat{L}D^{-1}, $$

where

$$ D = \begin{bmatrix} \|f_r\|_2I & 0 & 0 \\ 0 & \|f_g\|_2I & 0 \\ 0 & 0 & \|f_b\|_2I \end{bmatrix}, \quad \hat{L} = \begin{bmatrix} 2I + L & -I & -I \\ -I & 2I + L & -I \\ -I & -I & 2I + L \end{bmatrix}. $$

Therefore, the operation $R(\beta)f$ can be interpreted as applying the standard 3D Laplacian (with circulant boundary conditions imposed on the third dimension) on a scaled version of $f$, with the results scaled back afterwards.

Let

$$ \Omega = \begin{bmatrix} \alpha_rI & 0 & 0 \\ 0 & \alpha_gI & 0 \\ 0 & 0 & \alpha_bI \end{bmatrix}, $$

where $\alpha_r$, $\alpha_g$ and $\alpha_b$ are positive constants, and $I$ is the identity matrix of appropriate size. Denote the element-wise square root of the matrix $\Omega$ by $\sqrt{\Omega}$ (note that $\Omega$ is a diagonal matrix with positive diagonal elements). The Galatsanos et al. regularization technique for the least squares solution takes the following form:

$$ \min_f \|u - Hf\|_2^2 + \|\sqrt{\Omega}R(\beta)f\|_2^2. \quad (4.40) $$

Here the constants $\alpha_r$, $\alpha_g$ and $\alpha_b$ are the regularization parameters that control the degree of regularization. Notice that the regularization operator $R$ depends on the vector $\beta$, which further depends on the solution. In the literature, it is common practice to use an estimate of $\beta$ rather than the exact $\beta$. However, our algorithm will solve for $\beta$ as part of the problem.
4.4.2 Solving the RSTLS problem for color image restoration

In this section, we show that the RSTLS algorithm in previous sections can be extended to the problem of color image restoration, that is, solving (3.10). Our assumption about the blurring matrix is that our knowledge of \( H_{ij} \), for \( i, j \in \{r, g, b\} \), is precise, but that of \( w \) is inexact. However, the basic algorithms of this section can be easily modified to handle the case in which our knowledge of \( H_{ij} \) is also inexact, but the preconditioning technique proposed in the next section could be less effective.

Let

\[
\tilde{w} = [\tilde{w}_{rr} \quad \tilde{w}_{gr} \quad \tilde{w}_{br} \quad \tilde{w}_{rg} \quad \tilde{w}_{gg} \quad \tilde{w}_{bg} \quad \tilde{w}_{rb} \quad \tilde{w}_{gb} \quad \tilde{w}_{bb}]^T
\]

be the modified \( w \). From our assumption, it follows that the modified blurring matrix is parameterized by \( \tilde{w} \). Therefore, the regularized STLS problem for color image restoration takes the following form:

\[
\min_{f, \tilde{w}} \| H(\tilde{w})f - u \|_2^2 + \| \sqrt{\Omega R(\beta)f} \|_2^2 + \alpha_h \| w - \tilde{w} \|_2^2
\]  

(4.41)

The last term in this formula can be interpreted as a penalty term for the size of the error in \( w \), and \( \alpha_h \) can be interpreted as the regularization parameter used to control the degree of regularization on that error.

For simplicity, let

\[
z = \tilde{w} - w = [z_{rr} \quad z_{gr} \quad z_{br} \quad z_{rg} \quad z_{gg} \quad z_{bg} \quad z_{rb} \quad z_{gb} \quad z_{bb}]^T
\]

then (4.41) becomes

\[
\min_{f, z} \| H(w + z)f - u \|_2^2 + \| \sqrt{\Omega R(\beta)f} \|_2^2 + \alpha_h \| z \|_2^2.
\]  

(4.42)

Experimental results showed that this formula tends to yield a darker version of the original image, and a \( \tilde{w} \) whose elements are systematically bigger than those of \( w \) (see Figure 4.14). This way it keeps the residual norm almost unchanged, but reduces the size of the solution.

To overcome this problem, we add the following constraints to (4.41):

\[
\begin{align*}
z_{rr} + z_{gr} + z_{br} = 0, \\
z_{rg} + z_{gg} + z_{bg} = 0, \\
z_{rb} + z_{gb} + z_{bb} = 0.
\end{align*}
\]  

(4.43)
That is, we require the sum of the modified weights for each observed channel be the same as that of the original ones. Equation (4.43) is an expression of the conservation of intensity values in the color image.

This looks as if an unconstrained optimization problem has been turned into a constrained one. However, since

\[
\begin{align*}
  z_{br} &= -z_{rr} - z_{gr}, \\
  z_{bg} &= -z_{rg} - z_{gg}, \\
  z_{bb} &= -z_{rb} - z_{gb},
\end{align*}
\]

now the modified blurring matrix, \( \tilde{H} \), is parameterized by only six variables, \( z_{rr}, z_{gr}, z_{rg}, z_{gg}, z_{rb} \) and \( z_{gb} \). Let

\[
\hat{z} = [z_{rr} \ z_{gr} \ z_{rg} \ z_{gg} \ z_{rb} \ z_{gb}]^T,
\]

then \( \hat{z} \) contains all the information in \( z \). Therefore, the above constrained optimization problem can be restated as follows:

\[
\min_{f, \hat{z}} \|H(w + z)f - u\|_2^2 + \|\sqrt{\Omega} R(\beta)f\|_2^2 + \alpha_h \|z\|_2^2.
\] (4.44)

The last term in (4.44) can be expressed in terms of \( \hat{z} \). Note that \( z = \Gamma \hat{z} \), with

\[
\Gamma = diag(\hat{\Gamma}, \hat{\Gamma}, \hat{\Gamma}), \quad \hat{\Gamma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & -1 \end{bmatrix}.
\]

Therefore,

\[
\|z\|_2^2 = \hat{z}^T \Gamma^T \Gamma \hat{z}.
\]

It is easy to check that the matrix \( S \) defined as follows is symmetric positive definite and satisfies \( S^2 = \Gamma^T \Gamma \) where

\[
S = diag(\hat{S}, \hat{S}, \hat{S}), \quad \hat{S} = \begin{bmatrix} \sqrt{\frac{\sqrt{3}+1}{2}} & \sqrt{\frac{\sqrt{3}-1}{2}} \\ \sqrt{\frac{\sqrt{3}-1}{2}} & \sqrt{\frac{\sqrt{3}+1}{2}} \end{bmatrix}.
\] (4.45)

Therefore, \( \|z\|_2 = \|S\hat{z}\|_2 \). It follows that (4.44) is equivalent to

\[
\min_{f, \hat{z}} \|H(w + z)f - u\|_2^2 + \|\sqrt{\Omega} R(\beta)f\|_2^2 + \alpha_h \|S\hat{z}\|_2^2.
\] (4.46)
For convenience, let
\[ \Psi(f, \hat{z}) = \|H(w + z)f - u\|_2^2 + \|\sqrt{\Omega}R(\beta)f\|_2^2 + \alpha_h\|S\hat{z}\|_2^2, \]
then equation (4.46) becomes
\[ \min_{f, \hat{z}} \Psi(f, \hat{z}). \quad (4.47) \]

If \( \beta \) is constant, this problem can be solved much as Pruessner and O’Leary solved the single channel image deblurring problem in [56]. Since \( \beta \) is not constant, we propose to solve (4.46) as follows. At each iteration, we fix \( \beta \), approximate the cost function by a quadratic function and minimize that function. At the end of each iteration, we use the new value of \( f \) to update \( \beta \).

If \( \beta \) is fixed, then
\[
\Psi(f + \Delta f, \hat{z} + \Delta \hat{z}) = \|H(w + z + \Delta z)(f + \Delta f) - u\|_2^2 \\
+ \|\sqrt{\Omega}R(\beta)(f + \Delta f)\|_2^2 + \alpha_h\|S(\hat{z} + \Delta \hat{z})\|_2^2.
\]
Ignoring higher order terms, we get
\[
\Psi(f + \Delta f, \hat{z} + \Delta \hat{z}) \approx \|H(w + z)(f + \Delta f) + H(\Delta z)\Delta f - u\|_2^2 \\
+ \|\sqrt{\Omega}R(\beta)(f + \Delta f)\|_2^2 + \alpha_h\|S(\hat{z} + \Delta \hat{z})\|_2^2.
\]
Let
\[
\tilde{\Psi}(\Delta f, \Delta \hat{z}) = \|H(w + z)(f + \Delta f) + H(\Delta z)\Delta f - u\|_2^2 \\
+ \|\sqrt{\Omega}R(\beta)(f + \Delta f)\|_2^2 + \alpha_h\|S(\hat{z} + \Delta \hat{z})\|_2^2.
\]
When \( \beta \) is fixed, minimizing \( \tilde{\Psi} \) is a Gauss-Newton step for minimizing \( \Psi \).

Define the matrix \( Z \) as follows,
\[
Z = \begin{bmatrix} Z_1 & 0 & 0 \\ 0 & Z_2 & 0 \\ 0 & 0 & Z_3 \end{bmatrix},
\]
where
\[
Z_1 = [H_{rr}f_r - H_{rb}f_b, \ H_{gr}f_g - H_{br}f_b,], \\
Z_2 = [H_{rg}f_r - H_{bg}f_b, \ H_{gg}f_g - H_{bg}f_b,], \\
Z_3 = [H_{rb}f_r - H_{bb}f_b, \ H_{gb}f_g - H_{bb}f_b].
\]
Then $Z$ satisfies

$$H(\Delta z)f = Z\Delta \hat{z}.$$  \hfill (4.52)

Therefore, $\hat{\Psi}$ becomes

$$\hat{\Psi}(\Delta f, \Delta \hat{z}) = \|H(w + z)(f + \Delta f) + Z\Delta \hat{z} - u\|_2^2$$

$$+ \|\sqrt{\Omega R(\beta)}(f + \Delta f)\|_2^2 + \alpha_h\|S(\hat{z} + \Delta \hat{z})\|_2^2. $$  \hfill (4.53)

It follows that minimizing $\hat{\Psi}$ is equivalent to solving the following linear least squares system:

$$\min_{\Delta f, \Delta \hat{z}} \begin{bmatrix} Z & H(w + z) \\ \sqrt{\alpha_b S} & 0 \\ 0 & \sqrt{\Omega R(\beta)} \end{bmatrix} \begin{bmatrix} \Delta \hat{z} \\ \Delta f \end{bmatrix} = \begin{bmatrix} u - H(w + z)f \\ -\sqrt{\alpha_b S}\hat{z} \\ -\sqrt{\Omega R(\beta)}f \end{bmatrix}. $$  \hfill (4.54)

The STLS algorithm for color image restoration is summarized as follows.

<table>
<thead>
<tr>
<th>Algorithm 4.4</th>
</tr>
</thead>
</table>
| 1. Set $f$ to be the least squares solution, $\hat{z} = 0$  
2. Repeat  
(a) Solve (4.54) for $\Delta f, \Delta \hat{z}$.  
(b) Set $f = f + \Delta f$, $\hat{z} = \hat{z} + \Delta \hat{z}$.  
(c) construct $Z$ according to (4.51),  
(d) update $\beta$ according to (4.39).  
until $\|\Delta f\|, \|\Delta \hat{z}\| \leq \epsilon$. |

We have no proof that this algorithm converges, however, experimental results indicate that this algorithm converges very quickly provided that the regularization parameters are not too small. It is recommended that the least squares solution be used as the initial guess. However, it was noticed in our experimental results that the convergence of this algorithm is not sensitive to the initial guess.

Algorithm 4.4 solves (4.46) for fixed regularization parameters $\alpha_r$, $\alpha_g$, $\alpha_b$ and $\alpha_h$. The optimal values for these parameters depend on the noise level of the observed image and estimated point spread functions. For the least squares solution, Ng and Bose proposed an efficient way to find the optimal $[\alpha_r, \alpha_g, \alpha_b]$ in [46]. For the RSTLS solution, we propose to use the optimal $[\alpha_r, \alpha_g, \alpha_b]$ computed by the least squares method. In order to find the optimal $\alpha_h$, we use an approach that is similar to the approach employed by Chan and Wong in [18]. That is, we start with a small value for $\alpha_h$, and increase it until a satisfactory solution is found.
4.5 Preconditioning the linear least squares systems efficiently

In this section we present an efficient way to solve the linear least squares system (4.11), (4.27), (4.37) and (4.54). The direct method for solving linear least squares problems, the QR factorization, needs \( O(mn^2) \) flops, and it does not allow us to take advantage of the structure of the matrix. These drawbacks make it unacceptable for large systems. On the other hand, Krylov subspace methods such as CGLS [10] and LSQR [55] have been proven to be effective for large sparse or structured systems. The basic operation of the CGLS or LSQR algorithm is matrix-vector multiplication.

The convergence rate of the CGLS or LSQR algorithm depends on the condition of the system being solved. If the condition number of the system is large, then convergence may be slow.

A linear least squares problem \( \min \| b - Ax \|_2^2 \) can be solved by applying the regular CGLS or LSQR algorithm to the transformed system

\[
\min_{\hat{x}} \| b - \hat{A}\hat{x} \|_2^2
\]

where \( \hat{A} = AG^{-1}, \hat{x} = Gx \). We choose a matrix \( G \), called a preconditioner, such that \( \hat{A} \) is well conditioned. For the preconditioned LSQR algorithm, we need to solve two linear systems at each iteration, one is in the form of \( Gx = b \) and the other is in the form of \( G^T x = b \). For the preconditioned CGLS algorithm, we need to solve a linear system in the form of \( Mx = b \), where \( M = G^TG \), at each iteration. Here \( M \) can also be interpreted as the preconditioner for \( A^TA \). We remark that for the CGLS algorithm, there is no need to obtain an explicit form for \( G \).

In the rest of this section, our analysis focuses on preconditioning the least squares system that is encountered in the RSTLS algorithm for color image restoration, that is, equation (4.54). However, this analysis, in a slightly modified or easier form, is also valid for the systems encountered in the RSTLS algorithm for single channel image deblurring and high resolution image reconstruction. Furthermore, we assume that Neumann boundary conditions are used. A brief discussion on how to precondition (4.54) if zero boundary conditions are used will be given at the end of this section.

The coefficient matrix of the system that we are trying to precondition is

\[
A = \begin{bmatrix}
Z & H(w + z) \\
\sqrt{\alpha_l}S & 0 \\
0 & \sqrt{\Omega}R(\beta)
\end{bmatrix}.
\]
Thus we have

\[ A^T A = \begin{bmatrix} 
Z^T Z + \alpha h S^T S & Z^T H(w + z) \\
H(w + z)^T Z & H(w + z)^T H(w + z) + R(\beta)^T \Omega R(\beta) 
\end{bmatrix}. \]

Let \( \hat{M} \) be the lower right block of \( A^T A \), that is,

\[ \hat{M} = H(w + z)^T H(w + z) + R(\beta)^T \Omega R(\beta). \]

We remark that the inverse of \( \hat{M} \) can be easily computed if all the cross- and within-channel point spread functions are symmetric. By “symmetric”, we mean that \( h(x, y) = h(-x, y) = h(x, -y) = h(-x, -y) \). In fact, Ng and Bose efficiently computed the inverse of a matrix that is quite similar to \( \hat{M} \) in [46].

Ng et al. [47] showed that if the point spread function \( h \) is symmetric, then the corresponding blurring matrix \( H \) with Neumann boundary conditions can be diagonalized by the two dimensional Discrete Cosine Transform (DCT) [57] matrix \( C \). Therefore,

\[ H = \begin{bmatrix} 
C^T \Lambda_{rr} C & C^T \Lambda_{rg} C & C^T \Lambda_{rb} C \\
C^T \Lambda_{rg} C & C^T \Lambda_{gg} C & C^T \Lambda_{gb} C \\
C^T \Lambda_{rb} C & C^T \Lambda_{gb} C & C^T \Lambda_{bb} C 
\end{bmatrix}, \]

where \( \Lambda_{ij} \) (for \( i, j \in \{r, g, b\} \)) are real diagonal matrices.

The two dimensional Laplacian matrix with Neumann boundary conditions can also be diagonalized by the 2D DCT matrix. Let

\[ L = C^T \Lambda_L C, \]

then

\[ R = \begin{bmatrix} 
C^T (\Lambda_L + 2I) C & C^T (-\beta_{rg} I) C & C^T (-\beta_{rb} I) C \\
C^T (-\beta_{rg} I) C & C^T (\Lambda_L + 2I) C & C^T (-\beta_{gb} I) C \\
C^T (-\beta_{rb} I) C & C^T (-\beta_{gb} I) C & C^T (\Lambda_L + 2I) C 
\end{bmatrix}. \]

It follows that the matrix \( \hat{M} \) can be put into the following form:

\[ \hat{M} = (I \otimes C^T) \begin{bmatrix} 
\Lambda_{11} & \Lambda_{12} & \Lambda_{13} \\
\Lambda_{12} & \Lambda_{22} & \Lambda_{23} \\
\Lambda_{13} & \Lambda_{23} & \Lambda_{33} 
\end{bmatrix} (I \otimes C). \]
Here $\Lambda_{ij}$ are real diagonal matrices. The matrix with diagonal blocks can be permuted in to a block diagonal matrix so that $\hat{M}$ is expressed as

$$
\hat{M} = \tilde{C}^T \text{diag}(B_{1,1}, \ldots B_{1,n}, B_{2,1}, \ldots, B_{2,n}, \ldots, B_{n,1}, \ldots B_{n,n}) \tilde{C}
$$

(4.57)

where

$$
\tilde{C} = P(I \otimes C)
$$

for the appropriate permutation matrix $P$. Here each $B_{k,\ell}$ $(1 \leq k, \ell \leq n)$ is a $3 \times 3$ symmetric matrix, and we consider $n \times n$ image in each color channel. Next we show that $\hat{M}$ is symmetric and positive definite.

**Theorem 4.1.** Assume that $(w+z)_{ij} \geq 0$ for $i, j \in \{r, g, b\}$, and at least one of $(w+z)_{ri}$, $(w+z)_{gi}$ and $(w+z)_{bi}$ $(i \in \{r, g, b\})$ is nonzero. Then the matrix $\hat{M}$ is symmetric and positive definite.

**Proof.** It is clear that $\hat{M}$ is symmetric. According to (4.57), it suffices to show that $B_{k,\ell}$ are positive definite for $1 \leq k, \ell \leq n$. In fact, the matrices $B_{k,\ell}$ are given by

$$
B_{k,\ell} = Q_{k,\ell}(w+z)^T Q_{k,\ell}(w+z) + (U + [\Lambda_L]_{k\ell}I)^T \Phi^2 (U + [\Lambda_L]_{k\ell}I)
$$

where

$$
Q_{k,\ell}(w+z) = \begin{bmatrix}
(w+z)_{rr}[\Lambda_{rr}]_{k\ell} & (w+z)_{gr}[\Lambda_{gr}]_{k\ell} & (w+z)_{br}[\Lambda_{br}]_{k\ell} \\
(w+z)_{rg}[\Lambda_{rg}]_{k\ell} & (w+z)_{gg}[\Lambda_{gg}]_{k\ell} & (w+z)_{bg}[\Lambda_{bg}]_{k\ell} \\
(w+z)_{rb}[\Lambda_{rb}]_{k\ell} & (w+z)_{gb}[\Lambda_{gb}]_{k\ell} & (w+z)_{bb}[\Lambda_{bb}]_{k\ell}
\end{bmatrix},
$$

$$
U = \begin{bmatrix}
\frac{2}{\sqrt{\alpha_r}} & -\beta_{rg} & -\beta_{rb} \\
-\frac{1}{\beta_{rg}} & 2 & -\beta_{gb} \\
-\frac{1}{\beta_{rb}} & -\frac{1}{\beta_{gb}} & 2
\end{bmatrix}
$$

and

$$
\Phi = \begin{bmatrix}
\sqrt{\alpha_r} & 0 & 0 \\
0 & \sqrt{\alpha_g} & 0 \\
0 & 0 & \sqrt{\alpha_b}
\end{bmatrix}.
$$

Let

$$
F_{k,\ell} = \begin{bmatrix}
Q_{k,\ell}(w+z) \\
\Phi(U + [\Lambda_L]_{k,\ell}I)
\end{bmatrix}.
$$

(4.58)

It is clear that $B_{k,\ell}$ is positive definite if and only if $F_{k,\ell}$ has full column rank. We note that the eigenvalues of the discrete Laplacian with the Neumann boundary condition are given by

$$
[\Lambda_L]_{k\ell} = 4 \sin^2 \left( \frac{(k-1)\pi}{2n} \right) + 4 \sin^2 \left( \frac{(\ell-1)\pi}{2n} \right), \quad 1 \leq k, \ell \leq n.
$$
Therefore, we have, \([Λ_L]_{1,1} = 0\). It implies that \(F_{k,\ell}\) has full column rank for \(k\) and \(\ell\) except \(k = \ell = 1\).

Next we show that \(F_{1,1}\) also has full column rank. It is easy to check that the eigenvalues of \(U\) are 0, 3 and 3 and their corresponding eigenvectors are
\[
[β_{rb}, β_{gb}, 1]^T \quad [-β_{rb}, 0, 1]^T \quad [-β_{rg}, 1, 0]^T
\]
respectively. Therefore \(Φ(U + [Λ_L]_{k,\ell}I)x = 0\) if and only if \(k = \ell = 1\) and \(x\) is proportional to the nonnegative vector \([β_{rb}, β_{gb}, 1]^T\). Since discrete point spread functions are normalized such that their sums are equal to 1, it is easy to show that \([Λ_{ij}]_{1,1} = 1\). By the assumption, it is clear that \(Q_{1,1}(w + z)\) is a positive matrix, thus \(Q_{1,1}(w + z)x \neq 0\). Thus there is no vector \(x\) such that \(F_{1,1}x = 0\). It follows that \(F_{1,1}\) must have full column rank. The result follows.

The inverse of \(\hat{M}\) is given by
\[
\hat{M}^{-1} = (I \otimes C^T)P^T \text{diag}(B_{1,1}^{-1}, \ldots, B_{1,n}^{-1}, B_{2,1}^{-1}, \ldots, B_{2,n}^{-1}, \ldots, B_{n,1}^{-1}, \ldots, B_{n,n}^{-1})P(I \otimes C).
\]
Therefore, the inverse of \(\hat{M}\) can be computed by performing several 2D DCTs and computing the inverses of \(n^2 \times 3 \times 3\) matrices. The matrix \(\hat{M}\) can also be factored into \(\hat{M} = \hat{G}^T\hat{G}\) where
\[
\hat{G} = \text{diag}(G_{1,1}, \ldots, G_{1,n}, G_{2,1}, \ldots, G_{2,n}, \ldots, G_{n,1}, \ldots, G_{n,n})P(I \otimes C).
\]
and \(B_{k,\ell} = G_{k,\ell}^T G_{k,\ell}, 1 \leq k, \ell \leq n\) are the result of either the Cholesky factorization of \(B_{k,\ell}\) or the Q–R factorization of \(F_{k,\ell}\) in (4.58).

When all the cross- and within-channel point spread functions are symmetric, we propose to precondition \(A^T A\) by the matrix \(M\) defined as follows:
\[
M = \begin{bmatrix}
Z^T Z + \alpha_h S^T S & 0 \\
0 & \hat{M}
\end{bmatrix}.
\]
Since the eigenvalues of \(S\) are 1 and \(\sqrt{3}\) and \(\alpha_h > 0\), the matrix \(Z^T Z + \alpha_h S^T S\) is positive definite, and the preconditioner \(M\) is also positive definite. It is clear that \(A^T A\), when preconditioned by \(M\), is different from the identity matrix only at the first six rows and first six columns, which means it is the identity matrix plus a rank-12 change. Thus, excluding the effects of rounding, preconditioned CGLS is guaranteed to converge to the exact solution within 13 iterations. Since \(\hat{M}\) can be factored into \(\hat{M} = \hat{G}^T\hat{G}\), the preconditioner \(M\) can also be factored into \(M = \hat{G}^T G\), such that \(G\) can be used as the preconditioner for \(A\) in the preconditioned LSQR algorithm.

If not all the cross- and within-channel point spread functions are symmetric, we suggest constructing the preconditioner according to the symmetric part of the point spread functions. In [39], Kwan and Ng proved the following theorem and lemma.
Theorem 4.2. Let $h$ be an arbitrary point spread functions and $H_n$ be the blurring matrix of $h$ with the Neumann boundary conditions imposed. Then the optimal cosine transform preconditioner $c(H_n)$ of $H_n$ is the blurring matrix corresponding to the symmetric point spread function $h_s$ given by

$$h_s(i,j) = (h(i,j) + h(i,-j) + h(-i,j) + h(-i,-j))/4$$

with the Neumann boundary conditions imposed.

Lemma 4.1. Let $h$ be an arbitrary point spread function and $H_n$ be the blurring matrix of $h$ with the Neumann boundary conditions imposed. Define the distance between the point spread functions $h$ and $h_s$ as

$$\delta = \sum_{i,j} |h(i,j) - h_s(i,j)|.$$ 

Then

$$\|H_n - c(H_n)\|_1, \|H_n - c(H_n)\|_\infty \leq 4\delta,$$

where $c(H_n)$ is the optimal cosine transform preconditioner for $H_n$.

Here we consider the preconditioner for $A^T A$ is given by

$$M_c = \begin{bmatrix} c(H(w + z))^T c(H(w + z)) + R(\beta)^T \Omega R(\beta) & 0 \\ 0 & Z^T Z + \alpha h S^T S \end{bmatrix} \quad (4.59)$$

where

$$c(H(w + z)) = \begin{bmatrix} (w + z)_{rr} c(H_{rr}) & (w + z)_{gr} c(H_{gr}) & (w + z)_{br} c(H_{br}) \\ (w + z)_{rg} c(H_{rg}) & (w + z)_{gg} c(H_{gg}) & (w + z)_{bg} c(H_{bg}) \\ (w + z)_{rb} c(H_{rb}) & (w + z)_{gb} c(H_{gb}) & (w + z)_{bb} c(H_{bb}) \end{bmatrix}.$$ 

Using the above lemma, we have the following result. First recall that

$$\|w + z\|_\infty = \max_{i,j \in \{r,g,b\}} (w + z)_{ij}.$$ 

Lemma 4.2. Let $A$ be the matrix given in (4.56) and $M_c$ be the preconditioner given in (4.59). If

$$\max_{i,j \in \{r,g,b\}} \|H_{ij} - c(H_{ij})\|_{1,\infty} \leq 4\delta$$

then

$$\|H(w + z)^T H(w + z) - c(H(w + z))^T c(H(w + z))\|_2 \leq 8\delta\|w + z\|_\infty.$$
Proof. We note that

\[ \|H_{ij}^T H_{k\ell} - c(H_{ij})^T c(H_{k\ell})\| \leq \|H_{ij}^T[H_{k\ell} - c(H_{k\ell})]\| + \|H_{ij}^T c(H_{k\ell})\| \]

\[ \leq \|H_{ij}^T\|[H_{k\ell} - c(H_{k\ell})]\| + \|H_{ij}^T c(H_{k\ell})\|\||c(H_{k\ell})\| \]

with \(i, j, k, \ell \in \{r, g, b\}\). Since the discrete point spread function is normalized such that its sum is equal to 1,

\[ \|H_{ij}^T\|_1 = \|H_{ij}^T\|_\infty = \|c(H_{ij})\|_1 = \|c(H_{ij})\|_\infty = 1. \]

Using the above results and the statement (4.60), we can estimate

\[ \|H(w + z)^T H(w + z) - c(H(w + z))^T c(H(w + z))\|_{1, \infty} \leq 8\delta \|w + z\|_\infty. \]

Hence the result follows. \(\blacksquare\)

According to Lemma 4.2, we see that if the point spread function is close to symmetric, then \(M_c\) will be a good approximation for \(A^T A\). It follows that if we construct the matrix \(\tilde{M}\) according to the symmetric part of the point spread functions, then the matrix \(M_c\) defined as in (4.59) will be a good preconditioner for \(A^T A\).

To show that the spectra of the preconditioned matrices are clustered around 1, we consider a sequence of blurring matrices and investigate the spectra of the sequence of the preconditioned blurring matrices. Therefore, we assume that the given blurring matrix \(H_{ij}\) comes from a fixed singly-infinite matrix \(H_{ij}^\infty\). We associate with \(H_{ij}^\infty\) the function \(\phi_{ij}\) where the entries of \(H_{ij}\) and \(H_{ij}^\infty\) are the Fourier coefficients of \(\phi_{ij}\). The function \(\phi_{ij}\) is called the generating function of the sequence of blurring matrices, see [15] for details. With the generating function \(\phi_{ij}\) for the blurring matrix \(H_{ij}\), it can be shown that the eigenvalues of the matrix \(c(H_{ij})\) are given by

\[ \phi_{ij} \left( \frac{(k-1)\pi}{2n}, \frac{(\ell-1)\pi}{2n} \right) \quad 1 \leq k, \ell \leq n, \quad i, j \in \{r, g, b\} \]

and the largest eigenvalue of \(c(H_{ij})\) is given by \(\phi_{ij}(0,0)\) when \(\ell = k = 1\), see for instance [15, 39]. Since discrete point spread functions are normalized such that their sums are equal to 1, it is easy to show that \(\phi_{ij}(0,0) = 1\). Next we show that the smallest eigenvalue of \(M_c\) is uniformly bounded away from 0 by a positive constant independent of the size of \(M_c\).
THEOREM 4.3. Assume that \((w+z)_{ij} \geq 0\) for \(i, j \in \{r, g, b\}\), and at least one of \((w+z)_{ri}\), \((w+z)_{gi}\) and \((w+z)_{bi}\) \((i \in \{r, g, b\})\) is nonzero. Then there exists a positive scalar \(\gamma_{\min}\), independent of the size of \(M_c\), such that the smallest eigenvalue of \(M_c\) is uniformly bounded away from zero by \(\gamma_{\min}\).

Proof. For simplicity, define

\[
\xi(x, y) = \alpha_{\min}(F_{x,y}^T F_{x,y}),
\]

where

\[
F_{x,y} = \begin{bmatrix} Q_{x,y}(w+z) & \Phi(U + \psi(x,y)I) \\
\end{bmatrix},
\]

\[
Q_{x,y}(w+z) = \begin{bmatrix} (w+z)_{rr} \phi_{rr}(x,y) & (w+z)_{gr} \phi_{gr}(x,y) & (w+z)_{br} \phi_{br}(x,y) \\
(w+z)_{rg} \phi_{rg}(x,y) & (w+z)_{gg} \phi_{gg}(x,y) & (w+z)_{bg} \phi_{bg}(x,y) \\
(w+z)_{rb} \phi_{rb}(x,y) & (w+z)_{gb} \phi_{gb}(x,y) & (w+z)_{bb} \phi_{bb}(x,y) \\
\end{bmatrix},
\]

\[
\psi(x,y) = 4\sin^2 x + 4\sin^2 y.
\]

Using this form, we have shown in Theorem 4.1 that \(F_{x,y}\) has full column rank and hence the smallest eigenvalue of the matrix \(F_{x,y}^T F_{x,y}\) is positive. Therefore, \(\xi(x, y) > 0\) for all \((x, y) \in [0, \pi/2]^2\). Since \(\xi(x, y)\) is a continuous function on \([0, \pi/2]^2\), there exists a constant \(\gamma_{\min}\) independent of the size of the matrix such that \(\xi(x, y) \geq \gamma_{\min} > 0\) for all \((x, y) \in [0, \pi/2]^2\). Hence the result follows. \(\square\)

By using Lemma 4.2 and Theorem 4.3, we can prove the following theorem.

THEOREM 4.4. Assume that \((w+z)_{ij} \geq 0\) for \(i, j \in \{r, g, b\}\), and at least one of \((w+z)_{ri}\), \((w+z)_{gi}\) and \((w+z)_{bi}\) \((i \in \{r, g, b\})\) is nonzero. If

\[
\max_{i,j \in \{r,g,b\}} \|H_{ij} - c(H_{ij})\|_{1,\infty} \leq 4\delta,
\]

then

\[
M_c^{-1}A^T A = I + E_1 + E_2,
\]

(4.61)

where \(E_1\) is a matrix of rank-12, and \(\|E_2\|_2 \leq 8\delta\|w + z\|_{\infty}/\gamma_{\min}\). Here \(\gamma_{\min}\) is the positive constant given in Theorem 4.3.
Proof. We note that
\[
M_c^{-1} A^T A
= I + M_c^{-1} (A^T A - M_c)
= I + M_c^{-1} \begin{bmatrix}
H(w + z)^T H(w + z) - c(H(w + z))^T c(H(w + z)) & H(w + z)^T Z \\
Z^T H(w + z) & 0
\end{bmatrix}.
\]
Let
\[
E_1 = M_c^{-1} \begin{bmatrix}
0 & H(w + z)^T Z \\
Z^T H(w + z) & 0
\end{bmatrix},
\]
and
\[
E_2 = M_c^{-1} \begin{bmatrix}
(H(w + z)^T H(w + z) - c(H(w + z))^T c(H(w + z))) & 0 \\
0 & 0
\end{bmatrix}.
\]
Then $E_1$ is a matrix of rank-12, and
\[
\|E_2\|_2 \leq \|M_c^{-1}\|_2 \left\| \begin{bmatrix}
H(w + z)^T H(w + z) - c(H(w + z))^T c(H(w + z)) & 0 \\
0 & 0
\end{bmatrix} \right\|_2
\leq 8\delta \|M_c^{-1}\|_2 \|w + z\|_\infty \leq 8\delta \|w + z\|_\infty / \gamma_{min}.
\]
Therefore (4.61) is satisfied. \[\square\]

According to Theorem 4.4, if $\delta$ is sufficiently small (i.e., if the point spread function is quite symmetric), then the spectra of the preconditioned matrices $M_c^{-1} A^T A$ are clustered around 1. Therefore the preconditioned conjugate gradient method will converge very quickly. In the next section, this result is confirmed by our experimental results.

If zero boundary conditions rather than Neumann boundary conditions are used, then the blurring matrices will have a Block Toeplitz with Toeplitz Blocks (BTTB) structure. A BTTB matrix can be approximated by a Block Circulant with Circulant Blocks (BCCB) matrix \cite{16, 12}, which can be diagonalized by the two dimensional Fourier transform matrix \cite{67}. Therefore, we propose to precondition the system (4.54) with a preconditioner that is similar to what is proposed above, but based on Fast Fourier Transform (FFT) \cite{67} rather than DCT.

4.6 Experimental Results

In this section, we demonstrate the importance of the RSTLS algorithms for the problems of high resolution image reconstruction and color image restoration proposed in §4.3 and §4.4, and the effectiveness of the preconditioner proposed in §4.5.
4.6.1 Results on high resolution image reconstruction

The picture shown in Figure 4.1, is used for computer simulation. A $2 \times 2$ sensor array with sub-pixel displacement errors are used to retrieve four $125 \times 125$ noisy under-sampled images. The actual sub-pixel displacements of each sensor are as follows:

- Sensor #1: $x = 0.55, y = 0.55$
- Sensor #2: $x = 0.55, y = 0.45$
- Sensor #3: $x = 0.45, y = 0.55$
- Sensor #4: $x = 0.45, y = 0.45$

The estimated sub-pixel displacements are set to be the ideal displacement, therefore, the error in the estimate is as follows:

- Sensor #1: $x = 0.05, y = 0.05$
- Sensor #2: $x = 0.05, y = -0.05$
- Sensor #3: $x = -0.05, y = 0.05$
- Sensor #4: $x = -0.05, y = -0.05$

The observed high resolution image is shown in Figure 4.2.

Neumann boundary conditions are used to build the blurring matrix, and the Laplacian is used as the regularization operator.

Figure 4.3 shows the L-curve of the regularized least squares (RLS) method and the RSTLS method. Here the residual norm of the RSTLS method is measured as
\[ \sqrt{\|r(u,v,f)\|^2 + \|u\|^2 + \|v\|^2} \] . As can be seen, the RSTLS method yields a smaller residual norm and a smaller size of the solution, which means the solution satisfies the system of equations better and has a smaller size. This figure also shows that if we choose to minimize the residual norm subject to the size of the solution is smaller than some threshold value, the RSTLS method will give a smaller residual norm than the RLS method.

The optimal regularization parameter for the RLS method is $\alpha = 0.010$, and the reconstructed image is shown in Figure 4.4. The optimal regularization parameter for the RSTLS method is $\alpha = 0.006$, and the reconstructed image is shown in Figure 4.5. As can be seen from these pictures, the image reconstructed by the RSTLS method exhibits better image quality, particularly the edges are sharper than the image reconstructed by the RLS method.

By using the optimal regularization parameter, the displacement errors recovered by the RSTLS method are as follows:

- Sensor #1: $x = 0.0571, y = 0.0447$
- Sensor #2: $x = 0.0570, y = -0.0435$
- Sensor #3: $x = -0.0279, y = 0.0447$
- Sensor #4: $x = -0.0267, y = -0.0436$
As can be seen, these numbers agree with the actual displacement errors well.

Numerical results also showed that the convergence property of Algorithm 4.3 is very good. The error level (measured as \( \|\Delta u^T \Delta v^T \Delta f^T\|^2 \)) of this algorithm at each outer iteration for selected regularization parameters is shown in Table 4.1. As can be seen, for all regularization parameters used in this test, Algorithm 4.3 needs no more than 6 iterations to converge to an error level of \(10^{-2}\).

<table>
<thead>
<tr>
<th>Iteration</th>
<th>(\alpha = 0.02)</th>
<th>(\alpha = 0.01)</th>
<th>(\alpha = 0.006)</th>
<th>(\alpha = 0.003)</th>
<th>(\alpha = 0.001)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.9638</td>
<td>7.3472</td>
<td>9.2905</td>
<td>11.9172</td>
<td>12.5794</td>
</tr>
<tr>
<td>2</td>
<td>0.5763</td>
<td>0.6220</td>
<td>0.6994</td>
<td>0.9037</td>
<td>1.5844</td>
</tr>
<tr>
<td>3</td>
<td>0.1552</td>
<td>0.2379</td>
<td>0.3504</td>
<td>0.5356</td>
<td>0.6323</td>
</tr>
<tr>
<td>4</td>
<td>0.0512</td>
<td>0.0571</td>
<td>0.0709</td>
<td>0.0862</td>
<td>0.0727</td>
</tr>
<tr>
<td>5</td>
<td>0.0097</td>
<td>0.0128</td>
<td>0.0186</td>
<td>0.0262</td>
<td>0.0197</td>
</tr>
<tr>
<td>6</td>
<td>0.0049</td>
<td>0.0046</td>
<td>0.0046</td>
<td>0.0052</td>
<td>0.0046</td>
</tr>
</tbody>
</table>

Figure 4.6 shows the error level at each CGLS iteration during the first outer iteration when the optimal regularization parameter is used. As can be seen, the CGLS algorithm needed only 23 iterations to converge when the preconditioning technique described in the previous section is used. However, without any preconditioning, it does not converge after 200 iterations. This demonstrates the effectiveness of the preconditioning technique.
Fig. 4.1. The original high resolution image

Fig. 4.2. The observed high resolution image
Fig. 4.3. The L-curve of the RLS and RSTLS method

Fig. 4.4. Reconstructed image by the RLS method
Fig. 4.5. Reconstructed image by the RSTLS method

Fig. 4.6. Error at each CGLS iteration during first outer iteration of Algorithm 4.3, \( \alpha = 0.006 \)
4.6.2 Results on color image restoration

The picture shown in Figure 4.7 is used for computer simulation.

![Image](image.png)

**Fig. 4.7.** The original color image

The discretized point spread functions are as follows.

\[
\begin{align*}
\mathbf{h}_{rr} &= \mathbf{h}_{gr} = \mathbf{h}_{br} = \begin{bmatrix}
\frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\
\frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\
\frac{1}{9} & \frac{1}{9} & \frac{1}{9}
\end{bmatrix}, \\
\mathbf{h}_{rg} &= \mathbf{h}_{gg} = \mathbf{h}_{bg} = \begin{bmatrix}
0 & 0.125 & 0 \\
0.125 & 0.5 & 0.125 \\
0 & 0.125 & 0
\end{bmatrix}, \\
\mathbf{h}_{rb} &= \mathbf{h}_{gb} = \mathbf{h}_{bb} = \begin{bmatrix}
0.0625 & 0.125 & 0.0625 \\
0.125 & 0.25 & 0.125 \\
0.0625 & 0.125 & 0.0625
\end{bmatrix}.
\end{align*}
\]

The weights used to generate the observed channels are as follows:

\[
\mathbf{w} = \begin{bmatrix} 0.5 & 0.3 & 0.2 & 0.25 & 0.5 & 0.25 & 0.2 & 0.3 & 0.5 \end{bmatrix}^T.
\]
The noise returned by the MATLAB [26] command “0.01*(rand-0.5)” is added to the observed channels. Figure 4.8 shows the observed image.

The estimated weights are set to

\[ \mathbf{w} = [0.45 \ 0.33 \ 0.22 \ 0.275 \ 0.45 \ 0.275 \ 0.22 \ 0.33 \ 0.55]^T. \]

The relative error on the estimated weights is 10%.

Two least squares solutions are computed. One is obtained by using the \( \mathbf{\beta} \) calculated from the observed image, the other is obtained by using the \( \mathbf{\beta} \) calculated from the original image. Figure 4.9 and 4.10 show these two images. For both solutions, \( [\alpha_r \ \alpha_g \ \alpha_b] = [0.0001 \ 0.0001 \ 0.0001] \) is used.

For the TLS solution, \( \alpha_h = 9 \) yields satisfactory results. The recovered weights are as follows.

\[ \mathbf{w} = [0.4943 \ 0.3137 \ 0.1920 \ 0.2555 \ 0.4688 \ 0.2757 \ 0.1902 \ 0.3292 \ 0.4807]^T. \]

The relative error of the recovered weights is 5.4%, which is much smaller than that of the original estimates. Figure 4.11 shows the image recovered by the STLS method. It can be seen that the visual difference between the TLS solution and least squares solutions is very obvious. The PSNR of the images also indicates that the TLS solution is much better.
Figure 4.12 shows the error norm (measured as $\|\Delta f^T \Delta \hat{z}^T\|$) at each iteration of Algorithm 4.4. As can be seen, Algorithm 4.4 converged to an error level of $10^{-2}$ in 12 iterations. More iterations will further reduce the error norm, but will have very little effect on the recovered image.

Two variants of Algorithm 4.4 are also used to compute TLS solutions. One uses a fixed $\beta$, which is calculated from the observed image. The image recovered by this variation is shown in Figure 4.13. The other variant does not apply the constraints stated in (4.43). That is, the solution of (4.42) is computed. The image recovered by this variant is shown in Figure 4.14. The calculated PSNR of these images indicate they are inferior to the image obtained by Algorithm 4.4.

Experiments also showed that the preconditioning technique proposed in §4.5 is very effective if the point spread functions are symmetric or close to symmetric. As predicted by theoretical analysis, preconditioned CGLS converges to the exact solution in 13 iterations when the point spread functions are symmetric. When the point spread functions are not symmetric, there is no guarantee that preconditioned CGLS will converge to the exact solution in 13 iterations, but the convergence rate is considerably accelerated if the functions are close to symmetric. This is shown in Figure 4.15. In this example, a noise returned by the MATLAB command “0.01*rand” is added to the elements of the original symmetric point spread functions. The point spread functions are then normalized such that they integrate to one. As can be seen, preconditioned CGLS converges much faster.
Fig. 4.9. The image recovered by the least squares method. The observed image is used to calculate $\beta$. PSNR=27.79

Fig. 4.10. The image recovered by the least squares method. The original image is used to calculate $\beta$. PSNR=28.01
Fig. 4.11. The image recovered by Algorithm 4.4, PSNR=34.22

Fig. 4.12. Error norm at each iteration of Algorithm 4.4
Fig. 4.13. The image recovered by a variant of Algorithm 4.4 (a fixed $\beta$ is used), PSNR=33.40

Fig. 4.14. The image recovered by a variant of Algorithm 4.4 (without the constraints stated in (4.43)), PSNR=29.47
Fig. 4.15. Relative residual for the normal equations at each CGLS iteration during first outer iteration of Algorithm 4.4.
Chapter 5

LAD and LMN for image restoration

In this chapter we consider the LAD and LMN solutions for the problem of single channel image deblurring. The techniques developed in this chapter can be applied to the problem of high resolution image reconstruction with little modification. Recall that the LAD solution can be stated as

\[
\min_f \|g - Hf\|_1 + \alpha \|Rf\|_1, \tag{5.1}
\]

and that the LMN solution can be stated as

\[
\min_f \|g - Hf\|_2^2 + \alpha \|Rf\|_1. \tag{5.2}
\]

It is worth pointing out that the term \(\|Rf\|_1\) is closely related to total variation regularization in image restoration. In the one dimensional case, if \(R\) is chosen to be the first order difference operator, then \(\|Rf\|_1\) is the discrete version of the total variation norm. For the details about the total variation, we refer to [60, 40].

Problems (5.1) and (5.2) are not easy to solve because the objective functions are non-smooth [5, 11]. For instance, Mamone [42] solved an \(\ell 1\) approximation problem with a non-negative solution using linear programming techniques, and Abdelmalek [2, 1] solved the problem as a quadratic programming problem whose solution is non-negative and bounded. However, they studied the algebraic image restoration problem in the one-dimensional only, for example, separable blurring functions. Pruessner and O’Leary [56] considered minimization of total least \(\ell 1\) norm by using linear programming for image restoration. However, since they simply use the MATLAB [26] function `linprog.m` to solve the linear programming problem, their algorithm was restricted to very small images. In [21, 22, 23, 24], Dax investigated regularized least norm problems with using the identity matrix as the regularization operator (i.e., \(R = I\)), and developed row-relaxation methods for solving such least norm problems. The row-relaxation method is suitable for solving problems in which \(H\) is sparse. On the other hand, Li and Santosa [40] employed both the steepest descent and an affine scaling Newton method to solve the
total variation image restoration problem, and Combettes [19] used an adaptive level set method for non-differentiable constrained image recovery. An alternative formulation in which total variation is used as a constraint in a general convex programming framework was proposed by Combettes and Pesquet in [20].

The main contribution of this chapter is to develop efficient algorithms for the minimization of mixed $\ell_2$-$\ell_1$ and $\ell_1$-$\ell_1$ norms for image restoration problems. We first formulate (5.1) as a linear programming problem and (5.2) as a quadratic programming problem. Then we solve these problems by combining an interior point method and a preconditioned conjugate gradient method, and show that factorized sparse inverse preconditioners are quite effective for minimization of mixed $\ell_2$-$\ell_1$ and $\ell_1$-$\ell_1$ norms in image restoration problems.

5.1 The LAD solution

5.1.1 Formulating the problem as a linear programming problem

Usually, images satisfy

$$f \geq 0,$$

that is, pixels have non-negative values. However, this constraint is often omitted in restoration methods, mainly because of the numerical intricacies it entails. The algorithms in this chapter provide the option to take into account this constraint.

Problem (5.1), with the non-negativity constraint imposed, can be formulated as a linear programming problem as follows. Little modification is needed if the non-negativity constraint is not desired.

Let $u = Hf - g$, let $v = \alpha Rf$. We split $u$ and $v$ into their non-negative and non-positive parts. That is, $u = u^+ - u^-$ and $v = v^+ - v^-$, where $u^+ = \max(u, 0)$, $u^- = \max(-u, 0)$, $v^+ = \max(v, 0)$, and $v^- = \max(-v, 0)$. The problem can now be written as

$$\min_{f, u^+, u^-, v^+, v^-} 1^T u^+ + 1^T u^- + 1^T v^+ + 1^T v^-$$

subject to

$$Hf - g = u^+ - u^-$$

$$\alpha Rf = v^+ - v^-$$

$$u^+, u^-, v^+, v^- \geq 0, f \geq 0$$

We use $1$ to denote the vector of all ones of appropriate size. This notation is used throughout the rest of this chapter.
Clearly (5.3) is a linear programming problem in the standard form:

$$\min_{x} \ c^T x \quad \text{subject to} \quad A x = b, \quad x \geq 0,$$

(5.4)

where $A$, $b$, $c$ and $x$ are defined as follows.

$$A = \begin{bmatrix} H & -I & I & 0 & 0 \\ \alpha R & 0 & 0 & I & 0 \end{bmatrix}, \quad b = \begin{bmatrix} g \\ 0 \end{bmatrix}.$$

$$x = \begin{bmatrix} f \\ u^+ \\ u^- \\ v^+ \\ v^- \end{bmatrix} \quad \text{and} \quad c = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

The Lagrangian function for (5.4) is

$$L(x, \lambda, s) = c^T x - \lambda^T (Ax - b) - s^T x.$$

(5.5)

Here $\lambda$ and $s$ are the Lagrange multiplier vectors for the constraints $Ax = b$ and $x \geq 0$ respectively. For clarity, we partition $\lambda$ as

$$\lambda = \begin{bmatrix} \lambda_u \\ \lambda_v \end{bmatrix},$$

(5.6)

where $\lambda_u$ is the Lagrange multiplier vector for the constraint $Hf - u^+ + u^- = g$, and $\lambda_v$ is for $\alpha R - v^+ + v^- = 0$.

### 5.1.2 Interior point methods

Primal-dual interior point methods have become a common choice for solving large linear programming problems. We briefly outline our adapted interior point method below. A detailed description of interior point methods for linear programming can be found in [68, Chapter 1] or [53, Chapter 14].

The optimality condition for (5.4) is as follows.

$$F(x, \lambda, s) = \begin{bmatrix} A^T \lambda + s - c \\ Ax - b \\ XS1 \end{bmatrix} = 0, \quad x \geq 0, \quad s \geq 0,$$

(5.7)

where

$$X = \text{diag}(x) \quad \text{and} \quad S = \text{diag}(s).$$

(5.8)
Primal-dual interior point methods have their origin in Newton’s method for the system of nonlinear equations (5.7). Newton’s method starts with some initial guess for the solution, and calculates a search direction at each iteration by solving a linearized model of the original system. A detailed description of Newton’s method for nonlinear systems can be found in [53, Chapter 11]. In the primal-dual interior point algorithm, the basic Newton step is modified such that the search directions are aimed at points on the central path \((x_\tau, \lambda_\tau, s_\tau)\), defined as

\[
F(x_\tau, \lambda_\tau, s_\tau) = \begin{bmatrix}
0 \\
0 \\
\tau 1
\end{bmatrix}, \quad x_\tau > 0, \quad s_\tau > 0.
\] (5.9)

Very often \(\tau\) is written as \(\sigma \mu\), where \(\sigma \in [0, 1]\) is a centering parameter, and \(\mu\) is the duality measure defined by

\[
\mu = \frac{1}{n} \sum_{i=1}^{n} x_i s_i = \frac{x^T s}{n}.
\] (5.10)

The step length at each iteration is chosen such that the new iterate is strictly positive, that is, \(x > 0\) and \(s > 0\).

The Newton search direction, \((\Delta x, \Delta \lambda, \Delta s)\), is computed by solving the following linear system:

\[
\begin{bmatrix}
0 & A^T & I \\
A & 0 & 0 \\
S & 0 & X
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta \lambda \\
\Delta s
\end{bmatrix} =
\begin{bmatrix}
-r_c \\
r_b \\
r_a
\end{bmatrix},
\] (5.11)

where

\[
r_c = A^T \lambda + s - c, \quad r_b = Ax - b, \quad r_a = XS1 - \sigma \mu 1.
\]

By eliminating \(\Delta s\) in (5.11), we obtain

\[
\begin{bmatrix}
-X^{-1} S & A^T \\
A & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta \lambda
\end{bmatrix} =
\begin{bmatrix}
-\hat{r}_c \\
-r_b
\end{bmatrix},
\] (5.12)

where \(\hat{r}_c = r_c - X^{-1} r_a\). Let \(D = S^{-1/2} X^{1/2}\), then (5.12) can be written as

\[
\begin{bmatrix}
-D^{-2} & A^T \\
A & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta \lambda
\end{bmatrix} =
\begin{bmatrix}
-\hat{r}_c \\
-r_b
\end{bmatrix}.
\] (5.13)

Note that since the components of \(x\) and \(s\) are strictly positive, all the diagonal elements of \(D\) are well defined.
From here the standard approach is to reduce (5.13) to the following form and solve it for the step $\Delta \lambda$:

$$AD^2A^T \Delta \lambda = -AD^2 \hat{r}_c - r_b.$$  

(5.14)

The application of the preconditioned MINRES [54] or the preconditioned GMRES [62] to (5.12) may be considered. There have been some works on preconditioning systems in the form of (5.12). For example, we could use the preconditioner of Benzi and Golub [7] which is briefly described in §4.

However, after comparing the performance of the different implementations, we found it advantageous to proceed as follows. The system (5.13) can be written as

$$
\begin{bmatrix}
-D_1^{-2} & 0 & 0 & 0 & 0 & H^T & \alpha R^T \\
0 & -D_2^{-2} & 0 & 0 & 0 & -I & 0 \\
0 & 0 & -D_3^{-2} & 0 & 0 & I & 0 \\
0 & 0 & 0 & -D_4^{-2} & 0 & 0 & -I \\
0 & 0 & 0 & 0 & -D_5^{-2} & 0 & I \\
H & -I & I & 0 & 0 & 0 & 0 \\
\alpha R & 0 & 0 & -I & I & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\Delta f \\
\Delta u^+ \\
\Delta u^- \\
\Delta v^+ \\
\Delta v^- \\
\Delta \lambda_u \\
\Delta \lambda_v
\end{bmatrix}
= 
\begin{bmatrix}
-\hat{r}_{c1} \\
-\hat{r}_{c2} \\
-\hat{r}_{c3} \\
-\hat{r}_{c4} \\
-\hat{r}_{c5} \\
-r_{b1} \\
-r_{b2}
\end{bmatrix}
$$

(5.15)

Here $D_i$ (for $i = 1, 2, 3, 4, 5$) are sub-matrices of $D$ of appropriate size, and $\hat{r}_{ci}$ (for $i = 1, 2, 3, 4, 5$) and $r_{bi}$ (for $i = 1, 2$) are sub-vectors of appropriate size of $\hat{r}_c$ and $r_b$, respectively.

By eliminating $\Delta u^+, \Delta u^-, \Delta v^+$ and $\Delta v^-$, we obtain

$$
\begin{bmatrix}
-D_1^{-2} & H^T & \alpha R^T \\
H & D_2^2 + D_3^2 & 0 \\
\alpha R & 0 & D_4^2 + D_5^2
\end{bmatrix}
\begin{bmatrix}
\Delta f \\
\Delta \lambda_u \\
\Delta \lambda_v
\end{bmatrix}
= 
\begin{bmatrix}
-\hat{r}_{c1} \\
-\hat{r}_{b1} \\
-\hat{r}_{b2}
\end{bmatrix},
$$

(5.15)

where

$$\hat{r}_{b1} = r_{b1} - D_2^2 \hat{r}_{c2} + D_3^2 \hat{r}_{c3}, \quad \text{and} \quad \hat{r}_{b2} = r_{b2} - D_4^2 \hat{r}_{c4} + D_5^2 \hat{r}_{c5}.$$  

By eliminating $\Delta \lambda_u$ and $\Delta \lambda_v$ from (5.15), we obtain

$$[D_1^{-2} + H^T(D_2^2 + D_3^2)^{-1}H + \alpha^2 R^T(D_4^2 + D_5^2)^{-1} R] \Delta f = \hat{r}_{c1},$$  

(5.16)

where

$$\hat{r}_{c1} = \hat{r}_{c1} - H^T(D_2^2 + D_3^2)^{-1} \hat{r}_{b1} - \alpha R^T(D_4^2 + D_5^2)^{-1} \hat{r}_{b2}.$$
The system (5.16) can be viewed as the normal equations for a linear least squares problem with the following coefficient matrix:

\[
\begin{bmatrix}
D_1^{-1} & (D_2^2 + D_3^2)^{-1/2}H \\
\alpha(D_4^2 + D_5^2)^{-1/2}R & (D_4^2 + D_5^2)^{-1/2}
\end{bmatrix}
\begin{bmatrix}
I \\
H
\end{bmatrix}.
\]

(5.17)

Clearly the second factor in the right hand side of (5.17) has full column rank and is well conditioned. Since the matrices \(D_i\) are all diagonal matrices with positive diagonal elements, the first factor in the right hand side of (5.17) has full column rank. It follows that the coefficient matrix in (5.16) is symmetric positive definite. Therefore, the system (5.16) can be solved by the conjugate gradient method.

In general, when the iterates are close to the solution, if the primal variable is close to zero, then the corresponding dual variable will be far away from zero. This means the corresponding element in \(D\) will be very small. On the other hand, if the primal variable is far away from zero, then the corresponding dual variable will be very close to zero, leading to a large entry in \(D\). Therefore, when the iterates are close to the solution, the matrix \(D\) will have both huge and tiny elements, causing ill conditioning. In §5.3, we discuss how to precondition the system (5.16) efficiently.

Once we have solved (5.16) for \(\Delta f\), the other unknowns in (5.11) can be recovered by the following equations:

\[
\begin{align*}
\Delta \lambda_u &= (D_2^2 + D_3^2)^{-1}(-\hat{r}_{i1} - H\Delta f), \\
\Delta \lambda_v &= (D_4^2 + D_5^2)^{-1}(-\hat{r}_{i2} - \alpha R\Delta f), \\
\Delta u^+ &= D_2^2(\hat{r}_{c2} + \Delta \lambda_u), \\
\Delta u^- &= D_2^2(\hat{r}_{c3} + \Delta \lambda_u), \\
\Delta v^+ &= D_4^2(\hat{r}_{c4} - \Delta \lambda_v), \\
\Delta v^- &= D_4^2(\hat{r}_{c5} + \Delta \lambda_v), \\
\Delta s &= -r_c - A^T\Delta \lambda.
\end{align*}
\]

(5.18)

5.1.3 Finding an initial feasible point and staying feasible

Define the strictly feasible set \(\mathcal{F}^0\) as

\[
\mathcal{F}^0 = \{(x, \lambda, s)|Ax = b, A^T\lambda + s = c, x > 0, s > 0\}.
\]

(5.19)
Finding an initial strictly feasible point is not straightforward in many linear programming problems. For our problem, an initial strictly feasible point can be found as follows. Let $\epsilon$ be some positive value. Let $f_\text{ls}$ be any estimate of the sought-after solution, e.g. the least squares solution. We choose $f^{(0)}$ to be

$$f^{(0)}[i] = \max\{f_\text{ls}[i], \epsilon\}.$$ 

We let $u^+(0)$ be the sum of $\epsilon$ and the non-negative part of $Hf^{(0)}$, and $u^-(0)$ be the sum of $\epsilon$ and the non-positive part of $Hf^{(0)}$. Then the vector

$$x^{(0)} = \begin{bmatrix} f^{(0)} \\ u^+(0) \\ u^-(0) \\ v^+(0) \\ v^-(0) \end{bmatrix}$$

satisfies $Ax^{(0)} = b$. Furthermore, $x^{(0)} \geq \epsilon$. There are many ways that we can find $\lambda^{(0)}$ and $s^{(0)}$ such that $s^{(0)} > 0$ and $A^T\lambda^{(0)} + s^{(0)} - c = 0$. Since our focus is on the problem where $H$ corresponds to a shift invariant blurring operator, we can let $\lambda^{(0)}_u$ be a constant vector of $-0.5$, and let $\lambda^{(0)}_v$ be the zero vector. Assuming Neumann boundary conditions are used, and the blurring function is normalized such that it integrates to one, Then $H^T\lambda^{(0)}_u$ will be a constant vector of $-0.5$. Therefore, if we let $s^{(0)} = c - A^T\lambda^{(0)}$, then $s^{(0)} \geq 0.5$, and $(x^{(0)}, \lambda^{(0)}, s^{(0)})$ is strictly feasible. If zero boundary conditions are used, this procedure would still give an initial strictly feasible point under weak assumptions on the blurring function.

If the current iterate is in the strictly feasible set, and (5.11) is solved exactly, and that a step length is chosen such that the new iterate satisfies $x > 0$ and $s > 0$, then the new iterate is strictly feasible.

However, when we use the conjugate gradient method to solve (5.16), only an approximate solution is given. Therefore, even if the current iterate is strictly feasible, the new iterate may not satisfy the conditions

$$Ax = b \quad \text{and} \quad A^T\lambda + s - c = 0$$

to very high accuracy. Thus, if no precaution is taken, the subsequent iterates could move further from the feasible set due to error propagation.

Fortunately, it is easy to show that if (5.18) is used to recover the other unknowns after (5.16) is solved, the conditions $Ax = b$ and $A^T\lambda + s - c = 0$ will be satisfied to machine precision even if (5.16) is solved inexactly. Therefore, if a proper step length is chosen, the next iterate will be strictly feasible.
5.1.4 Summary of the adapted interior point algorithm

There are many variants of the interior point method. Our algorithm, summarized
below, is based on the simplest framework, that is, Framework PD in [68, page 8].

Algorithm: Interior point for the LAD solution
1. Initialize \((x^{(0)}, \lambda^{(0)}, s^{(0)}) \in \mathcal{F}^o\).
2. for \(k = 0, 1, 2 \ldots\)
   
   choose \(\sigma \in [0, 1]\).
   
   solve (5.16) for \(\Delta f\).
   
   Use (5.18) to recover \(\Delta u^+, \Delta u^-, \Delta v^+, \Delta v^-, \Delta \lambda, \Delta s\).
   
   choose \(\beta^{pri}\) and \(\beta^{dual}\).
   
   \(x^{(k+1)} = x^{(k)} + \beta^{pri} \Delta x\)

   \(\lambda^{(k+1)} = \lambda^{(k)} + \beta^{dual} \Delta \lambda\)

   \(s^{(k+1)} = s^{(k)} + \beta^{dual} \Delta s\)
end

Note that no predictor-corrector strategy is involved in our algorithm. This is
because we use iterative methods to solve the linear systems.

A word needs to be said about the choices of centering parameter \(\sigma\) and step
lengths \(\beta^{pri}\) and \(\beta^{dual}\). It is common practice that the primal and dual step lengths \(\beta^{pri}\)
and \(\beta^{dual}\) be chosen as follows:

\[
\beta^{pri} = \min(1, \eta \beta^{pri}_{max}), \quad \beta^{dual} = \min(1, \eta \beta^{dual}_{max}).
\]

Here \(\beta^{pri}_{max}\) and \(\beta^{dual}_{max}\) are the maximum step that can be taken before violating the non-
negativity condition, and \(\eta \in [0.9, 1.0]\). We simply choose \(\eta = 0.95\). Our experiments
led us to choose the centering parameter \(\sigma\) to be 0.1 at the first iteration, and

\[
\sigma = \max(0.01, (1 - \min(\beta^{pri}, \beta^{dual}))^{3})
\]

in other iterations. This prevents the iterates from moving too far apart from the central
path and allows significant reduction in duality at each iteration.

5.2 The LMN solution

Problem (5.2), with the non-negativity constraint imposed, can be restated as
a quadratic programming problem as follows. We let \(v = \alpha R f\), and split \(v\) into its
non-negative and non-positive parts \(v^+\) and \(v^-\). Then (5.2) can be written as

\[
\min_{f, v^+, v^-} 1^T v^+ + 1^T v^- + \|Hf - g\|^2_2
\]  \hspace{1cm} (5.20)
subject to

$$\alpha R f = v^+ - v^-$$

$$v^+, v^-, f \geq 0.$$ 

Problem (5.20) can be written as

$$\min_x \frac{1}{2} x^T G x + c^T x \quad \text{subject to} \quad A x = b, \quad x \geq 0,$$

where $G, A, b, c$ and $x$ are defined as follows.

$$G = \begin{bmatrix} 2H^T & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad A = \begin{bmatrix} \alpha R & -I & I \end{bmatrix},$$

$$b = 0, \quad x = \begin{bmatrix} f \\ v^+ \\ v^- \end{bmatrix} \quad \text{and} \quad c = \begin{bmatrix} -2H^T g \\ 1 \\ 1 \end{bmatrix}.$$ 

We remark that the objective function is convex since $G$ is symmetric positive semi-definite. Problem (5.21) can also be solved by the interior point method. The Lagrangian function for (5.21) is

$$L(x, \lambda, s) = \frac{1}{2} x^T G x + c^T x - \lambda^T (A x - b) - s^T x,$$

where $\lambda$ and $s$ are the generalized Lagrange multiplier vectors for the constraints $A x = b$ and $x \geq 0$ respectively.

The optimality condition for (5.21) is

$$F(x, \lambda, s) = \begin{bmatrix} G x + c - A^T \lambda - s \\ A x - b \\ X S 1 \end{bmatrix} = 0, \quad x \geq 0, \quad s \geq 0.$$

The central path, $(x_{\sigma \mu}, \lambda_{\sigma \mu}, s_{\sigma \mu})$, is defined similarly as in the previous section:

$$F(x_{\sigma \mu}, \lambda_{\sigma \mu}, s_{\sigma \mu}) = \begin{bmatrix} 0 \\ 0 \\ \sigma \mu 1 \end{bmatrix}. \quad (5.24)$$
Recall that $\sigma \in (0, 1)$ and $\mu$ defined as in (5.10). The Newton step, $(\Delta x, \Delta \lambda, \Delta s)$, is computed by solving the following linear system:

$$
\begin{bmatrix}
G & -A^T & -I \\
A & 0 & 0 \\
S & 0 & X
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta \lambda \\
\Delta s
\end{bmatrix}
= \begin{bmatrix}
-r_c \\
r_b \\
r_a
\end{bmatrix},
$$

(5.25)

where

$$
r_c = Gx + c - A^T \lambda - s, \quad r_b = Ax - b, \quad r_a = XS1 - \sigma \mu 1.
$$

By eliminating $\Delta s$, we obtain

$$
\begin{bmatrix}
G + X^{-1}S & -A^T \\
A & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta \lambda
\end{bmatrix}
= \begin{bmatrix}
-\hat{r}_c \\
r_b
\end{bmatrix},
$$

(5.26)

where $\hat{r}_c = r_c + X^{-1}r_a$. This system can be made symmetric by a change of variable:

$$
\begin{bmatrix}
G + X^{-1}S & A^T \\
A & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
-\Delta \lambda
\end{bmatrix}
= \begin{bmatrix}
-\hat{r}_c \\
r_b
\end{bmatrix}.
$$

(5.27)

Let $D = S^{-1/2}X^{1/2}$. Note that the elements of $D$ are well defined. Partition $D$ as

$$
D = \text{diag}(D_1, D_2, D_3),
$$

where $D_1, D_2$ and $D_3$ are matrices of appropriate size such that the system (5.27) can be put into the following form:

$$
\begin{bmatrix}
2H^TH + D_1^{-2} & 0 & 0 & \alpha R^T \\
0 & D_2^{-2} & 0 & -I \\
\alpha R & 0 & D_3^{-2} & I
\end{bmatrix}
\begin{bmatrix}
\Delta f \\
\Delta v^+ \\
\Delta v^- \\
-\Delta \lambda
\end{bmatrix}
= \begin{bmatrix}
-\hat{r}_{c1} \\
-\hat{r}_{c2} \\
-\hat{r}_{c3} \\
-r_b
\end{bmatrix}.
$$

(5.28)

Here $\hat{r}_{ci}$ (for $i = 1, 2, 3$) are sub-vectors of appropriate size of $\hat{r}_c$. By eliminating $\Delta v^+$ and $\Delta v^-$, we get:

$$
\begin{bmatrix}
2H^TH + D_1^{-2} & \alpha R^T \\
\alpha R & -D_2^{-2} - D_3^{-2}
\end{bmatrix}
\begin{bmatrix}
\Delta f \\
-\Delta \lambda
\end{bmatrix}
= \begin{bmatrix}
-\hat{r}_{c1} \\
-r_b
\end{bmatrix},
$$

(5.29)

where

$$
\hat{r}_b = r_b + D_2^2\hat{r}_{c2} - D_3^2\hat{r}_{c3}.
$$
Eliminating $\triangle \lambda$, we get

$$
[2H^T H + D_1^{-2} + \alpha^2 R^T (D_2^2 + D_3^2)^{-1} R] \triangle f = -\tilde{r}_{c1}, \quad (5.30)
$$

where

$$
\tilde{r}_{c1} = \hat{r}_{c1} + \alpha R^T (D_2^2 + D_3^2)^{-1} \hat{r}_b.
$$

Similar to the system (5.16), the system (5.30) can be viewed as the normal equations of a linear least squares problem, whose coefficient matrix has full column rank. Therefore, the coefficient matrix in (5.30) is symmetric positive definite, and the system (5.30) can be solved by the preconditioned conjugate gradient method.

Once (5.30) has been solved, the other unknowns can be calculated by the following equations:

$$
\begin{align*}
\triangle \lambda &= (D_2^2 + D_3^2)^{-1} (-\hat{r}_b - \alpha R \triangle f), \\
\triangle v^+ &= D_2^2 (\hat{r}_{c2} - \triangle \lambda), \\
\triangle v^- &= D_2^2 (\hat{r}_{c3} + \triangle \lambda), \\
\triangle s &= G \triangle x - A^T \triangle \lambda + r_c.
\end{align*}
$$

(5.31)

An initial feasible point can be found similarly as was done for the LAD solution. Moreover, if (5.31) is used to recover the rest of the unknowns after (5.30) is solved approximately, and a proper step length is chosen, then the iterates will stay feasible.

The interior point algorithm for the LMN solution is quite similar to that for the LAD solution. One difference that is worth pointing out is that $\beta^{pri}$ and $\beta^{dual}$ are allowed to take different values in the algorithm for the LAD solution but have to be the same in that for the LMN solution.

### 5.3 Preconditioning the structured linear systems

As mentioned earlier, the linear systems (5.16) and (5.30) become ill conditioned as the iterates get close to the solution. Thus preconditioners are needed to accelerate the convergence of the conjugate gradient iterations.

For the non-negative least squares image restoration problem, Saunders [63] used preconditioned LSQR to solve the linear system at each interior point iteration. He used a diagonal preconditioner. However, the effectiveness of diagonal preconditioners is limited, in that preconditioned LSQR needed more than one thousand iterations to converge to a desired level of accuracy at the final stages of the interior point algorithm. Our experimental results indicate that the conditions of the systems (5.16) and (5.30) tend to be worse than that of the systems in the non-negative least squares problem. This should be no surprise since our problems are transformed from non-smooth problems.
5.3.1 Saddle Point Preconditioners

For the linear systems (5.12), Benzi and Golub [7] proposed using the preconditioned GMRES to solve the equivalent system

\[
\begin{bmatrix}
D^{-2} & -A^T \\
A & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta \lambda
\end{bmatrix} =
\begin{bmatrix}
\hat{r}_c \\
-r_b
\end{bmatrix}.
\]  

(5.32)

Their preconditioning strategy is based on the symmetric/skew-symmetric splitting of the coefficient matrix, that is, they propose using the following matrix as the preconditioner:

\[
M_\gamma = \begin{bmatrix}
D^{-2} + \gamma I & 0 \\
0 & \gamma I
\end{bmatrix}
\begin{bmatrix}
\gamma I & -A^T \\
A & \gamma I
\end{bmatrix}.
\]  

(5.33)

Here \( \gamma \) is some positive constant. Linear systems in the form of \( M_\gamma z = r \) can be solved by first solving

\[
\begin{bmatrix}
D^{-2} + \gamma I & 0 \\
0 & \gamma I
\end{bmatrix}
\tilde{z} = r,
\]  

(5.34)

followed by

\[
\begin{bmatrix}
\gamma I & -A^T \\
A & \gamma I
\end{bmatrix} z = \tilde{z}.
\]  

(5.35)

Solving (5.34) is a simple dot-wise division. Since our focus is on shift invariant blur, the system (5.35) can be solved by the discrete cosine transform if the blur is symmetric and Neumann boundary conditions are applied, see [47] for instance. This preconditioning technique can also be applied to (5.15), which yields better results in practice.

5.3.2 Factorized Sparse Inverse Preconditioner

Our experimental results indicate that the most efficient way to compute the Newton step is to apply the Factorized Sparse Inverse Preconditioner (FSIP) [37, 8, 65] to the systems (5.16) and (5.30). Let \( B \) be a symmetric positive definite matrix, and its Cholesky factorization be \( B = CC^T \). The idea of FSIP is to find the lower triangular matrix \( L \) with sparsity pattern \( S \) such that

\[
\| I - CL \|_F
\]

is minimized, where \( \| \cdot \|_F \) denotes Frobenius norm. Kolotilina and Yeremin [37] showed that \( L \) can be obtained by the following algorithm:
Algorithm: Construction of FSIP

1. Compute $\hat{L}$ with sparsity pattern $S$ such that $[\hat{L}B]_{ij} = \delta_{ij}$, $(i,j) \in S$.
2. Let $\hat{D} = (\text{diag}(\hat{L}))^{-1}$ and $L = \hat{D}^{1/2}\hat{L}$.

According to this algorithm, $n$ small linear systems need to be solved, where $n$ is the number of rows in the matrix $B$. These systems can be solved in parallel, and is well suited for modern massive parallel computing.

A special type of FSIP, the Factorized Banded Inverse Preconditioner (FBIP) was proposed by Lin et al. [41] for Toeplitz related matrices. We say that $B \in \mathbb{R}^{m \times n}$ has lower bandwidth $p$ if $b_{ij} = 0$ whenever $i > j + p$. The upper bandwidth of a matrix is defined similarly. Note that by this definition, a tridiagonal matrix has upper and lower bandwidth of 1.

Lin et al. [41] proved the following theorem.

**Theorem 5.1.** Let $T$ be a Hermitian Toeplitz matrix. Denote the $k$-th diagonal of $T$ by $t_k$. Assume the diagonals of $T$ satisfy

\[ |t_k| \leq ce^{-\gamma |k|} \] (5.36)

for some $c > 0$ and $\gamma > 0$, or

\[ |t_k| \leq c(|k| + 1)^{-s} \] (5.37)

for some $c > 0$ and $s > 3/2$. Then for any given $\epsilon > 0$, there exists $p' > 0$ such that for all $p > p'$,

\[ \|L_p - C^{-1}\| \leq \epsilon, \]

where $L_p$ is the FBIP of $T$ with the lower bandwidth $p$, and $C$ is the Cholesky factor of $T$.

This theorem indicates that if a Toeplitz matrix $T$ has certain off-diagonal decay property, then the FBIP will be a good approximation to $T^{-1}$. Note that if a Toeplitz matrix is banded, then both (5.36) and (5.37) are satisfied trivially.

Lin et al. [41] also considered Toeplitz-related systems of the form $I + TT^* DT$, where $D$ is a positive diagonal matrix, and proved the following corollary.

**Corollary 5.1.** Let $T$ be a Toeplitz matrix with its diagonals satisfying (5.36) or (5.37). Let $D = \text{diag}(d_i)$ with $0 < d_i \leq d$, and $B = I + TT^* DT$. 


Then for any given \( \epsilon > 0 \), there exists \( p' > 0 \) such that for all \( p > p' \),

\[
\|L_p - C^{-1}\| \leq \epsilon,
\]

where \( L_p \) is the FBIP of \( B \) with the lower bandwidth \( p \), and \( C \) is the Cholesky factor of \( B \).

The matrices we are trying to precondition are the coefficient matrices in (5.16) and (5.30), that is, \( B = D_1^{-2} + H^T(D_2^2 + D_3^2)^{-1}H + \alpha^2 R^T(D_4^2 + D_5^2)^{-1}R \) and \( B = 2H^T H + D_1^{-2} + \alpha^2 R^T(D_2^2 + D_3^2)^{-1}R \). While the focus of this research is on two-dimensional images, we have the following lemma for the one dimensional problem.

**Lemma 5.1.** Let \( T \) be a Toeplitz with its diagonals satisfying (5.36) or (5.37). Let \( D_1, D_2 \) and \( D_3 \) be diagonal matrices with positive diagonal entries. Let

\[
B = T^T D_1 T + D_2 + R^T D_3 R,
\]

be a well-conditioned matrix, where \( R \) is the first-order or the second-order difference operator. Then for any given \( \epsilon > 0 \), there exists \( p' > 0 \) such that for all \( p > p' \),

\[
\|L_p - C^{-1}\| \leq \epsilon,
\]

where \( L_p \) is the FBIP of \( B \) with lower bandwidth \( p \), and \( C \) is the Cholesky factor of \( B \).

**Proof.** The proof is similar to the proof of Corollary 4 in [41].

Lemma (5.1) indicates that the linear system in the one-dimensional LMN and LAD problems can be efficiently preconditioned by the FBIP.

It is important to note that in order to construct the FBIP for \( B \), it is not necessary to explicitly calculate all the elements of \( B \) when \( p \) is small. The following lemma indicates that only certain diagonals of \( B \) need to be calculated.

**Lemma 5.2.** Let \( L_p \) be the FBIP with lower bandwidth \( p \) for a symmetric matrix \( B \). Then \( L_p \) can be determined by the first \( p \) bands of the lower triangular part of \( B \).

**Proof.** For \( i = 1, 2, \ldots, p - 1 \), the nonzero elements of the \( i \)-th row of \( L_p \) is \( L(i, 1 : i) \). They are determined by \( B(1 : i, 1 : i) \).

On the other hand, for \( i = p, p + 1, \ldots, n \), the nonzero elements of the \( i \)-th row of \( L_p \) is \( L(i, i + 1 - p : i) \). They are determined by \( B(i + 1 - p : i, i + 1 - p : i) \).
Therefore, $L_p$ can be determined by the first upper and lower $p$ bands of $B$. Since $B$ is symmetric, the result follows.

For two-dimensional problems, we assume that the blurring matrix $H$ has a block-level off-diagonal decay property, and each block also has off-diagonal decay property. This is true if the blurring function decays in spatial domain, or if the support of the blur is small. In this case, we can set the FSIP to be a triangular block banded matrix with each block being a banded matrix. Let $p$ be the block-level lower bandwidth of the FSIP, and $q$ be the lower and the upper bandwidths of each block. Figure 5.1(a) shows the sparsity pattern of an FSIP with $p = 2$ and $q = 1$. Note that if $p = q = 0$, then the FSIP becomes a diagonal preconditioner.

Similar to the one dimensional case, when constructing the FSIP, it is not necessary to calculate all the elements of the coefficient matrix in (5.16) or (5.30). However, the situation for the two dimensional case is a little bit more complicated.

**Lemma 5.3.** Let $L$ be the FSIP of the coefficient matrix in (5.16) or (5.30), let $p$ be the block-level lower bandwidth and $q$ be the upper and the lower bandwidths of the blocks of $L$. Then $L$ can be determined by those elements that are in the lower triangular part of the coefficient matrix and have a block-level lower bandwidth of $p$, with each block having upper and lower bandwidth of $2q$.

Lemma 5.3 is actually a bit conservative. Figure 5.1(b) shows the sparsity pattern of the elements of the coefficient matrix that needs to be calculated if the FSIP has a sparsity pattern shown in Figure 5.1(a). Note that the blocks in the last block diagonal have lower bandwidth of $q$. If the original blurring matrix $H$ is sparse enough such that there is no need to use the FFT [67] for matrix multiplications, then the cost of calculating these elements is no more than $(p + 1)(2q + 1)$ unpreconditioned conjugate gradient iterations.

Other than explicitly calculating certain diagonals of the coefficient matrix in (5.16) or (5.30), the cost of constructing the FSIP consists of calculating the lower triangular factor $L$. Since this step solves $mn$ linear systems of size $(p + 1)(2q + 1) \times (p + 1)(2q + 1)$, the complexity of this step is $O(p^3 q^3 mn)$. Because this cost grows cubically with the increase of $p$ and $q$, the benefit of the decrease in the number of iterations will be offset by the cost of constructing the preconditioner with the increase of $p$ and $q$. Therefore, it is important to choose suitable values of $p$ and $q$ in practical applications.

Usually the linear system to be solved is well-conditioned at the first few interior point iterations. Due to the initial cost of constructing the FSIP, it is not worthwhile to choose a big $p$ or $q$. As the iterates get close to the solution, the condition number of the system get worse, and it becomes worth it to use bigger $p$ and $q$. Therefore, the best strategy is to choose small $p$ and $q$ (say $p = q = 0$) at the beginning, and increase $p$ and $q$ gradually. For details, see experimental results in the next section.
Fig. 5.1. (a) FSIP with $p = 2$ and $q = 1$; (b) Sparsity pattern of the elements of the coefficient matrix that needs to be calculated.

5.4 Experimental results

Most of our experiments are based on the camera man image shown in Figure 5.2. The blurring function is chosen to be a two dimensional Gaussian,

$$h(i, j) = e^{-2(i/3)^2 - 2(j/3)^2},$$

truncated such that the function has a support of $7 \times 7$.

We generate two observed images. For the first one, a Gaussian white noise with variance 0.05 is added. For the second one, we randomly pick 50% of the pixels to be contaminated by noise, the type and variance of the noise is the same as in the first one. The observed images are shown in Figure 5.3. For the first observed image, since the noise is normal, we only consider the LMN solution. For the second one, we consider both the LMN solution and LAD solution.

Neumann boundary conditions are used to build the blurring matrix $H$. Figure 5.4 shows the least squares restoration of the two observed images respectively. The optimal regularization parameter is used.

The interior point algorithms are implemented in MATLAB, except the construction of the FSIP, which is written as a MATLAB callable C function. Recall that in order to construct the FSIP, $mn$ small linear systems need to be solved. Doing this in MATLAB would be slow because MATLAB loops are inefficient.

The LMN solution for the first observed image is shown in Figure 5.5. The PSNR of this image indicates it is superior to the least squares solution. It is interesting to
note that the LMN solution is quite different from the least squares solution. The least squares solution tends to smooth out the edges but under-smoothes flat areas. The LMN solution tends to sharpen the edges and makes flat areas even flatter.

The LAD and LMN solutions for the second observed image is shown in Figure 5.6. As can be seen, both solutions have a higher PSNR than the least squares solution. Furthermore, the LAD solution is superior to the LMN solution.

Similar to total variation deblurring [60], using the $\ell_1$ norm for the regularization term is very effective in recovering “blocky” images. Using the $\ell_1$ norm for the residual term yields very good results if only a small portion of the pixels are contaminated by noise. Figure 5.7(a) to 5.7(e) show an original “blocky” image, blurred image with 50% of the pixels contaminated by additive noise, least squares restoration, LMN solution, and LAD restoration.

Figure 5.8 and 5.9 show the convergence of the primal-dual interior point algorithm for the LMN and LAD solutions respectively. Theoretically, the interior point iterations should exhibit super-linear convergence. The experimental results do not seem to confirm the theoretical predictions. This can be explained as follows. The linear systems are approximately solved such that the residual is below a certain level. Because the condition of the linear systems get worse as the iterates get closer to the solution, the error in the calculated search directions becomes bigger, causing poorer convergence. In fact, for small images, if direct methods are used to solve the linear systems, super-linear convergence is observed.

Table 5.1 shows the number of Conjugate Gradient iterations and CPU time at each interior point iteration for the LAD solution when different preconditioning
strategies are used. The CPU time listed here includes the time for constructing the preconditioners. Table 5.2 shows the same kind of results for the linear systems in the LMN solution. As can be seen, in both cases, the FSIP with proper choices of $p$ and $q$ can significantly speed up the convergence.

We remark that the performance of the FSIP depends on the support of the blurring function and how fast the function decays in space. The effect of the support of the blurring matrix on the effectiveness of the FSIP is two folded. On the one hand, a big support means a relatively dense blurring matrix, therefore allowing the preconditioner to be relatively dense. On the other hand, a big support usually means a relatively dense preconditioner is needed in order to improve the condition significantly. Table 5.3 shows the performance of the FSIP with different choice of $p$ and $q$ when the support of the blur is $5 \times 5$ (The same Gaussian blur is used). Table 5.4 shows the same kind of information when the support of the blur is $9 \times 9$.

In general, the faster the blurring function decays in space, the more effective the FSIP is. In that sense, the worst blurring function is a uniform blur. Table 5.5 shows the performance of the FSIP if a $7 \times 7$ uniform blur is used.

Fig. 5.3. Observed images: (a) all pixels are contaminated by noise; (b) only 50% of the pixels are contaminated by noise.
Table 5.1. No. of conjugate gradient iterations and CPU time needed at each interior point iteration for the LAD solution when different preconditioning strategies are used. The preconditioning strategies used include: (1) NOR, Eq. (5.14) is solved, using a diagonal preconditioner; (2) BG, the Benzi and Golub preconditioner is applied to (5.15); (3) No Pre, plain conjugate gradient is applied to (5.16); (4), Diag Pre, diagonal preconditioner is applied to (5.16); (5) FSIP2, FSIP with $p = q = 2$ is applied to (5.16); (6) FSIP3, FSIP with $p = q = 3$; (7) FSIP4, FSIP with $p = q = 4$.

<table>
<thead>
<tr>
<th>PD Itn</th>
<th>NOR</th>
<th>BG</th>
<th>No Pre</th>
<th>Diag Pre</th>
<th>FSIP2</th>
<th>FSIP3</th>
<th>FSIP4</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>&gt;3000</td>
<td>295</td>
<td>644</td>
<td>544</td>
<td>48</td>
<td>22</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-</td>
<td>161.78</td>
<td>78.68</td>
<td>68.72</td>
<td>15.24</td>
<td>13.32</td>
</tr>
<tr>
<td>12</td>
<td>&gt;3000</td>
<td>483</td>
<td>982</td>
<td>804</td>
<td>51</td>
<td>26</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-</td>
<td>237.70</td>
<td>125.92</td>
<td>105.53</td>
<td>15.75</td>
<td>14.14</td>
</tr>
<tr>
<td>14</td>
<td>&gt;3000</td>
<td>1047</td>
<td>1603</td>
<td>1334</td>
<td>60</td>
<td>31</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-</td>
<td>589.68</td>
<td>207.05</td>
<td>175.88</td>
<td>17.42</td>
<td>15.31</td>
</tr>
<tr>
<td>16</td>
<td>&gt;3000</td>
<td>1791</td>
<td>2432</td>
<td>1934</td>
<td>77</td>
<td>43</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-</td>
<td>969.84</td>
<td>305.14</td>
<td>242.51</td>
<td>20.90</td>
<td>18.47</td>
</tr>
<tr>
<td>18</td>
<td>&gt;2000</td>
<td>&gt;3000</td>
<td>&gt;2607</td>
<td>&gt;106</td>
<td>54</td>
<td>41</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-</td>
<td>339.06</td>
<td>25.85</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>&gt;2000</td>
<td>&gt;3000</td>
<td>&gt;3000</td>
<td>170</td>
<td>85</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-</td>
<td>37.71</td>
<td>28.18</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 5.2. No. of conjugate gradient iterations and CPU time needed at each interior point iteration for the LMN solution when different preconditioners are used. The preconditioning strategies used include: (1) No Pre, plain conjugate gradient is applied to (5.30); (2), Diag Pre, diagonal preconditioner is applied to (5.30); (3) FSIP2, FSIP with \( p = q = 2 \) is applied to (5.30); (4) FSIP3, FSIP with \( p = q = 3 \); (5) FSIP4, FSIP with \( p = q = 4 \).

<table>
<thead>
<tr>
<th>PD Itn</th>
<th>No Pre</th>
<th>Diag Pre</th>
<th>FSIP2</th>
<th>FSIP3</th>
<th>FSIP4</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>368</td>
<td>180</td>
<td>31</td>
<td>21</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>45.01</td>
<td>22.94</td>
<td>\textbf{6.15}</td>
<td>6.89</td>
<td>10.52</td>
</tr>
<tr>
<td>12</td>
<td>618</td>
<td>281</td>
<td>51</td>
<td>32</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>76.27</td>
<td>35.27</td>
<td>9.71</td>
<td>\textbf{9.24}</td>
<td>12.72</td>
</tr>
<tr>
<td>14</td>
<td>908</td>
<td>438</td>
<td>85</td>
<td>48</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>112.37</td>
<td>55.28</td>
<td>16.12</td>
<td>\textbf{13.35}</td>
<td>15.86</td>
</tr>
<tr>
<td>16</td>
<td>1391</td>
<td>626</td>
<td>139</td>
<td>71</td>
<td>49</td>
</tr>
<tr>
<td></td>
<td>171.36</td>
<td>78.27</td>
<td>25.95</td>
<td>\textbf{18.31}</td>
<td>20.84</td>
</tr>
<tr>
<td>18</td>
<td>2202</td>
<td>989</td>
<td>225</td>
<td>123</td>
<td>67</td>
</tr>
<tr>
<td></td>
<td>265.25</td>
<td>119.56</td>
<td>40.32</td>
<td>29.99</td>
<td>\textbf{26.55}</td>
</tr>
<tr>
<td>20</td>
<td>&gt;3000</td>
<td>1814</td>
<td>408</td>
<td>228</td>
<td>120</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>222.30</td>
<td>72.91</td>
<td>54.15</td>
<td>\textbf{41.62}</td>
</tr>
</tbody>
</table>

Table 5.3. No. of conjugate gradient iterations and CPU time needed at each interior point iteration for the LAD solution when different FSIP is used. The support of the blur is \( 5 \times 5 \).

<table>
<thead>
<tr>
<th>PD Itn</th>
<th>No Pre</th>
<th>Diag Pre</th>
<th>FSIP2</th>
<th>FSIP3</th>
<th>FSIP4</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>839</td>
<td>669</td>
<td>57</td>
<td>24</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>79.10</td>
<td>64.95</td>
<td>11.40</td>
<td>\textbf{9.39}</td>
<td>11.76</td>
</tr>
<tr>
<td>12</td>
<td>1406</td>
<td>1140</td>
<td>96</td>
<td>38</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>130.74</td>
<td>108.45</td>
<td>17.16</td>
<td>\textbf{11.94}</td>
<td>13.40</td>
</tr>
<tr>
<td>14</td>
<td>2199</td>
<td>1672</td>
<td>129</td>
<td>52</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>208.98</td>
<td>164.74</td>
<td>22.31</td>
<td>\textbf{14.61}</td>
<td>15.54</td>
</tr>
<tr>
<td>16</td>
<td>&gt;3000</td>
<td>2296</td>
<td>175</td>
<td>73</td>
<td>39</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>222.31</td>
<td>29.55</td>
<td>19.45</td>
<td>\textbf{18.64}</td>
</tr>
<tr>
<td>18</td>
<td>&gt;3000</td>
<td>&gt;3000</td>
<td>266</td>
<td>106</td>
<td>54</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>-</td>
<td>43.23</td>
<td>26.57</td>
<td>\textbf{22.80}</td>
</tr>
<tr>
<td>20</td>
<td>&gt;3000</td>
<td>&gt;3000</td>
<td>378</td>
<td>142</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>-</td>
<td>59.50</td>
<td>32.89</td>
<td>\textbf{28.19}</td>
</tr>
</tbody>
</table>
Table 5.4. No. of conjugate gradient iterations and CPU time needed at each interior point iteration for the LAD solution when different FSIP is used. The support of the blur is $9 \times 9$.

<table>
<thead>
<tr>
<th>PD Itn</th>
<th>No Pre</th>
<th>Diag Pre</th>
<th>FSIP2</th>
<th>FSIP3</th>
<th>FSIP4</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1184</td>
<td>578</td>
<td>74</td>
<td>46</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>198.76</td>
<td>100.03</td>
<td>26.66</td>
<td>24.41</td>
<td>25.43</td>
</tr>
<tr>
<td>12</td>
<td>1759</td>
<td>940</td>
<td>115</td>
<td>67</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>286.54</td>
<td>157.77</td>
<td>36.17</td>
<td>30.23</td>
<td>31.29</td>
</tr>
<tr>
<td>14</td>
<td>2394</td>
<td>1432</td>
<td>175</td>
<td>98</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>389.36</td>
<td>241.66</td>
<td>49.39</td>
<td>38.63</td>
<td>38.23</td>
</tr>
<tr>
<td>16</td>
<td>&gt;3000</td>
<td>2093</td>
<td>272</td>
<td>153</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>353.55</td>
<td>71.68</td>
<td>54.44</td>
<td>48.50</td>
</tr>
<tr>
<td>18</td>
<td>&gt;3000</td>
<td>&gt;3000</td>
<td>409</td>
<td>237</td>
<td>143</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>100.86</td>
<td>76.32</td>
<td>65.62</td>
<td>65.62</td>
</tr>
<tr>
<td>20</td>
<td>&gt;3000</td>
<td>&gt;3000</td>
<td>710</td>
<td>402</td>
<td>236</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>166.37</td>
<td>121.45</td>
<td>96.83</td>
<td>96.83</td>
</tr>
</tbody>
</table>

Table 5.5. No. of conjugate gradient iterations and CPU time needed at each interior point iteration for the LAD solution when different FSIP is used. The blur is a $7 \times 7$ uniform blur.

<table>
<thead>
<tr>
<th>PD Itn</th>
<th>No Pre</th>
<th>Diag Pre</th>
<th>FSIP2</th>
<th>FSIP3</th>
<th>FSIP4</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>454</td>
<td>346</td>
<td>96</td>
<td>94</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>57.25</td>
<td>44.83</td>
<td><strong>23.69</strong></td>
<td>30.28</td>
<td>38.72</td>
</tr>
<tr>
<td>12</td>
<td>731</td>
<td>535</td>
<td>118</td>
<td>113</td>
<td>105</td>
</tr>
<tr>
<td></td>
<td>92.19</td>
<td>69.10</td>
<td><strong>28.16</strong></td>
<td>33.76</td>
<td>45.07</td>
</tr>
<tr>
<td>14</td>
<td>1241</td>
<td>1022</td>
<td>169</td>
<td>156</td>
<td>141</td>
</tr>
<tr>
<td></td>
<td>158.31</td>
<td>135.20</td>
<td><strong>37.24</strong></td>
<td>44.61</td>
<td>55.51</td>
</tr>
<tr>
<td>16</td>
<td>1826</td>
<td>1596</td>
<td>250</td>
<td>229</td>
<td>197</td>
</tr>
<tr>
<td></td>
<td>221.76</td>
<td>199.87</td>
<td><strong>50.20</strong></td>
<td>61.90</td>
<td>72.77</td>
</tr>
<tr>
<td>18</td>
<td>2796</td>
<td>2306</td>
<td>334</td>
<td>287</td>
<td>269</td>
</tr>
<tr>
<td></td>
<td>347.30</td>
<td>295.07</td>
<td><strong>66.21</strong></td>
<td>75.45</td>
<td>92.30</td>
</tr>
<tr>
<td>20</td>
<td>&gt;3000</td>
<td>&gt;3000</td>
<td>550</td>
<td>464</td>
<td>411</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>-</td>
<td>104.73</td>
<td>117.31</td>
<td>136.25</td>
</tr>
</tbody>
</table>
Fig. 5.4. (a) The least squares solutions for the first observed image, PSNR = 20.46db, (b) The least squares solution for the second observed image, PSNR = 20.87db

Fig. 5.5. The LMN solution for the first observed image, PSNR = 20.78db
Fig. 5.6. (a) The LMN solution for the second observed image, PSNR = 21.30db, (b) The LAD solution for the second observed image, PSNR = 22.82db
Fig. 5.7.  (a) Original wheel image, (b) Observed noisy image, with 50% of the pixels contaminated by noise, (c) Least Squares restoration, PSNR=29.25 (d) The LMN solution, PSNR=30.64 (e) the LAD solution, PSNR=35.26
Fig. 5.8. Convergence of the primal-dual interior point method for the LMN solution: (a) Duality measure, (b) Infinity norm of $\Delta f$

Fig. 5.9. Convergence of the primal-dual interior point method for LAD solution: (a) Duality measure, (b) Infinity norm of $\Delta f$
6.1 Conclusion

In this thesis we explored new techniques for solving error contaminated ill-conditioned linear systems.

In Chapter 2, we showed the two regularization parameters of the Regularized TLS problem can be calculated by solving a system of nonlinear equations. The system of nonlinear equations can be efficiently solved by the well known Broyden’s method. Experimental results showed that our algorithm has superlinear convergence and outperforms existing algorithms. We also proposed an efficient way for plotting the L-curve.

Chapters 3, 4, and 5 focused on the problems of image restoration and high resolution image reconstruction. The mathematical models for single channel image restoration, high resolution image reconstruction, and color image restoration are given in Chapter 3.

In Chapter 4, we proposed RSTLS algorithms for high resolution image reconstruction and color image restoration. The RSTLS problems are stated as an unconstrained problems. Our algorithms solve the problems iteratively. At each iteration, the objective function is simplified by ignoring higher order terms. The simplified version of the objective function can be minimized by solving a linear least squares system. Experiments showed that our algorithms have very good convergence properties. Numerical results also showed that the RSTLS method recovers the error in the blurring matrix well, and yields an image that is superior to the least squares solution. We also proposed an efficient preconditioning technique for the linear system involved in our RSTLS algorithm. The efficiency of this preconditioner is demonstrated by examples.

Chapter 5 considered LAD and LMN for image restoration. The problems are formulated as solutions to smooth linear or quadratic programming problems, and solved by primal-dual interior point methods. The linear system at each interior point iteration is first reduced to a more compact system and then solved by the PCG method. The FSIP is used to speed up the convergence of the conjugate gradient iterations. Experimental results indicate that the LAD and LMN solutions recover the edges better than the least squares method. We also demonstrated that the FSIP can speed up the convergence considerably if a proper sparsity pattern is chosen.
6.2 Future work

6.2.1 Three dimensional image reconstruction

Another possible application of the techniques developed in this thesis is in transmission tomography [45]. For this problem, one probes an object with nondiffractive radiation. Let $I_0$ be the intensity of the radiation source, $f(x)$ be the linear attenuation coefficient of the object at the point $x$, and $L$ be the line along which the ray travels, and $I$ be the intensity of the radiation past the object, then

$$I = I_0 e^{-\int_L f(x) dx}, \quad (6.1)$$

or equivalently

$$\log \frac{I_0}{I} = \int_L f(x) dx. \quad (6.2)$$

In the ideal case, $\log \frac{I_0}{I}$ is directly measured and the problem of 3D reconstruction is to reconstruct $f(x)$ from a collection of measurements of $\log \frac{I_0}{I}$ along different directions. This can be modeled by a linear system

$$g_i = P_i f, \quad i = 1, 2, \ldots, M. \quad (6.3)$$

Here $P_i$ is the projection operator along the $i$th direction, it maps volume attenuation coefficients into a two-dimensional array, whose entries are simply the line integrals of the volume coefficient along the set of lines in the $i$th direction. $M$ is the number of projections. Usually the total number of data points exceeds the number of voxels (3D pixels) in the 3D structure, therefore, Eq.(6.3) constitutes an over-determined system.

Eq (6.3) is far from reality in many situations. In electron microscopy (see [69, 70]), for example, the projected image is a convolution of the line integrals with a point spread function (PSF), that is,

$$y_i = h_i * g_i, \quad i = 1, 2, \ldots, M, \quad (6.4)$$

or

$$y_i = h_i * (P_i f), \quad i = 1, 2, \ldots, M. \quad (6.5)$$

The PSF $h_i$ is the inverse Fourier transform of the Contrast Transfer Function (CTF). If we define $H_i$ as the two-dimensional convolution operator that corresponds to $h_i$, Eq.(6.5) can be written as

$$y_i = H_i P_i f, \quad i = 1, 2, \ldots, M. \quad (6.6)$$

Currently the most widely used method in the electron microscopy literature is regularized least squares. It would be very interesting to investigate if STLS, LMN and LAD performs better than least squares when applied to this problem.
6.2.2 Total least one norm or total least mixed norm for image restoration

The rationale behind TLS is that our knowledge of the coefficient matrix is inexact, and that behind LAD and LMN solution is that the prior distribution and/or the noise on the right hand side do not satisfy the Gaussian assumption. In reality, we often face situations where neither the Gaussian prior distribution or the assumption that exact information of the coefficient matrix is available is satisfied. Therefore, it would be reasonable to estimate the unknown parameters by solving

$$
\min_{\tilde{A},\tilde{x}} \| b - \tilde{A}x \|_2^2 + \| \tilde{A} - A \|_F^2 + \alpha \| Rx \|_1,
$$

(6.7)

or

$$
\min_{\tilde{A},\tilde{x}} \| b - \tilde{A}x \|_1 + \| \tilde{A} - A \|_1 + \alpha \| Rx \|_1.
$$

(6.8)

We call the solution to (6.7) the Total Least Mixed Norm solution and that of (6.8) the Total Least One Norm solution. In the case that the coefficient matrix is structured, then we compute the Structured Total Least Mixed Norm solution or the Structured Total Least One Norm solution, that is, we require the perturbed coefficient matrix have the same structure as the original matrix.
References


Vita

Haoying Fu

Haoying Fu was born in Hunan, China on September 26th, 1972. He received his Bachelor’s degree in Engineering Physics from Tsinghua University, China, in July 1994. He was employed at Shanghai Institute of Process Automation Instrumentation from August 1994 to July 1998. He received his Master’s degree in Nuclear Engineering from the Pennsylvania State University in August 2000. He began his doctoral studies in Computer Science and Engineering at the Pennsylvania State University in the fall of 2000. His current research interests are in the areas of numerical linear algebra and optimization, image processing, and data mining.