Technical Report

TR-2004-009

Quasi-Newton Methods for Image Restoration

by

James Nagy, Katrina Palmer

Mathematics and Computer Science

EMORY UNIVERSITY

Quasi-Newton Methods for Image Restoration

James G. Nagy* Katrina Palmer†

Abstract

Many iterative methods that are used to solve $A\mathbf{x} = \mathbf{b}$ can be derived as quasi-Newton methods for minimizing the quadratic function $\frac{1}{2}\mathbf{x}^T A^T A\mathbf{x} - \mathbf{x}^T A^T \mathbf{b}$. In this paper, several such methods are considered, including conjugate gradient least squares (CGLS), Barzilai-Borwein (BB), residual norm steepest descent (RNSD) and Landweber (LW). Regularization properties of these methods are studied by analyzing the so-called "filter factors". The algorithm proposed by Barzilai and Borwein is shown to have very favorable regularization and convergence properties. Secondly, we find that preconditioning can result in much better convergence properties for these iterative methods.

1 Introduction

We consider linear systems that arise from discretization of ill-posed problems,

$$\mathbf{b} = A\mathbf{x} + \mathbf{n}\,, \tag{1}$$

where A is an ill-conditioned $n \times n$ matrix, whose singular values decay to zero. The vector $\mathbf{b}$ and matrix A are assumed known, and the vector $\mathbf{n}$, which represents perturbations and/or noise in the data, is assumed unknown. The aim is to compute an approximation of the unknown vector, $\mathbf{x}$. Following the terminology of Hansen [11], we call such a linear system a *discrete ill-posed problem*. If $\mathbf{n}$ is small, it is tempting to ignore it, and use standard approaches to compute the *inverse solution*, $\mathbf{x} = A^{-1}\mathbf{b}$. However, due to the ill-posedness of the problem, this naive approach is likely to compute solutions that are horribly corrupted with noise. Techniques known as *regularization methods* have been developed to deal with this ill-posedness. In this paper we consider iterative methods that can be used to compute a regularized solution. More specifically, we show that a general quasi-Newton approach to solve $A\mathbf{x} = \mathbf{b}$ can be formulated as a filtering method for computing solutions of discrete ill-posed problems (1). We give an explicit formula for computing the quasi-Newton filter factors, and we show that the semi-convergent behavior of specific iterative methods (i.e., special cases of the general quasi-Newton method) is related to their corresponding filter factors.

We present the quasi-Newton method and several popular iterative methods derived from it in Section 2. In Section 3 we analyze the regularization properties of the methods described in Section 2. Section 4 will briefly describe the preconditioner we use for image restoration problems. Finally, Section 5 compares the filter factors of the four methods on two examples from image restoration. Convergence comparisons of the various methods (both with and without preconditioning) are also included.

*Math & CS Department, Emory University, Atlanta, GA 30322, USA. nagy@mathcs.emory.edu

†Department of Mathematical Sciences, Appalachian State University, Boone, NC 28608. kmp@cs.appstate.edu

2 Quasi-Newton Methods

Most iterative methods assume that the coefficient matrix is symmetric and positive definite. Because in our applications of interest A is not guaranteed to be symmetric, we consider approaches that solve the normal equations,

$$A^T A\mathbf{x} = A^T\mathbf{b}\,. \tag{2}$$

To compute a solution to (2), we can solve the minimization problem

$$\min f(x) = \frac{1}{2}\mathbf{x}^T A^T A\mathbf{x} - \mathbf{x}^T A^T\mathbf{b}\,. \tag{3}$$

For this quadratic function, we have

- gradient: $\mathbf{g}(\mathbf{x}) = \bigtriangledown f(\mathbf{x}) = A^T(A\mathbf{x} - \mathbf{b})$
- residual: $\mathbf{r} = A^T(\mathbf{b} - A\mathbf{x}) = -\mathbf{g}(\mathbf{x})$
- Hessian: $H(\mathbf{x}) = \bigtriangledown^2 f(\mathbf{x}) = A^T A$

To solve the minimization problem given in (3) we might consider using Newton's method [5]. This is derived using a second order Taylor series approximation of $f(\mathbf{x})$ near $\mathbf{x}_k$:

$$f(\mathbf{x}) = f(\mathbf{x}_k) + \bigtriangledown f(\mathbf{x}_k)^T(\mathbf{x} - \mathbf{x}_k) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_k)^T \bigtriangledown^2 f(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k)\,. \tag{4}$$

The right hand side of equation (4) is minimized at

$$\begin{aligned}\mathbf{x} &= \mathbf{x}_k - [\bigtriangledown^2 f(\mathbf{x}_k)]^{-1} \bigtriangledown f(\mathbf{x}_k) \\ &= \mathbf{x}_k - H(\mathbf{x}_k)^{-1}\mathbf{g}(\mathbf{x}_k)\,.\end{aligned}$$

This produces the basic Newton iteration :

$$\mathbf{x}_{k+1} = \mathbf{x}_k - H(\mathbf{x}_k)^{-1}\mathbf{g}(\mathbf{x}_k)\,. \tag{5}$$

Notice that in the case when $f(\mathbf{x})$ is quadratic (assuming A is nonsingular), we obtain

$$\mathbf{x}_{k+1} = \mathbf{x}_k - (A^T A)^{-1}A^T(A\mathbf{x}_k - \mathbf{b}) = A^{-1}\mathbf{b}. \tag{6}$$

So Newton's method converges to $A^{-1}\mathbf{b}$ in one iteration no matter what we choose for an initial guess, $\mathbf{x}_0$. However, we do not want to use Newton's method for two reasons. First, $A^{-1}\mathbf{b}$ is too corrupted with noise when solving $\mathbf{b} = A\mathbf{x} + \mathbf{n}$. For ill-posed problems we would instead prefer to have $\mathbf{x}_k$ converge to $A_r^\dagger\mathbf{b}$, a *regularized solution*. We use the notation $A_r^\dagger\mathbf{b}$ to denote the operator that maps $\mathbf{b}$ to a regularized solution, and therefore we may think of it as a regularized pseudo-inverse of A. We discuss how to reduce the effects of the noise when we talk about regularization in Section 3. Secondly, the point of using iterative methods is to avoid computing A^{-1}, or even some factorization of A . To address this problem we consider a quasi-Newton approach defined by:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k M_k\mathbf{g}_k = \mathbf{x}_k + \alpha_k M_k\mathbf{r}_k \tag{7}$$

where $\mathbf{g}_k = \mathbf{g}(\mathbf{x}_k)$, $\mathbf{r}_k = A^T(\mathbf{b} - A\mathbf{x}_k)$, and $\alpha_k M_k \approx A_r^\dagger$.

There are many different quasi-Newton methods but they are all based on approximating the inverse Hessian by another matrix, $\alpha_k M_k$ that is found at a lower cost. Some common examples include the following.

Example 1: If we let $\alpha_k = \alpha$ and $M_k = I$ be fixed with $0 \leq \alpha \leq \frac{2}{\sigma_1}$ then this leads to Landweber's iteration or Richardson's method [11]. Landweber's method (LW) is not used much in practice because the convergence rate is often very slow [11]. LW is a classic approach for ill-posed problems and its convergence properties are fairly easy to analyze.

Example 2: If we let $M_k = I$ be fixed, and choose $\alpha_k = \min_{\alpha>0} f(\mathbf{x}_k + \alpha \mathbf{r}_k)$ then we get the residual norm steepest descent method (RNSD) [17]. Note that RNSD is just steepest descent on the normal equations. Steepest descent minimizes the A^TA-norm of the error so RNSD minimizes $||\mathbf{x} - \mathbf{x}_k||_{A^TA} = ||A(\mathbf{x} - \mathbf{x}_k)||_2 = ||\mathbf{b} - A\mathbf{x}_k||_2$. Like LW, RNSD is typically not recommended due to its slow convergence. However, we show in Section 5 (see also [15]) that in some cases, with proper preconditioning, RNSD is competitive and exhibits a more stable convergence behavior than conjugate gradient least squares (CGLS).

Example 3: If we let $M_k = I$ be fixed, and choose $\alpha_k = \min_{\alpha>0} f(\mathbf{x}_{k-1} + \alpha \mathbf{r}_k)$ then we get the Barzilai-Borwein (BB) method [1]. Notice that the BB method is the same as RNSD except that BB uses the previous steepest descent step length. The algorithm exhibits some interesting convergence behavior. Specifically, BB (without preconditioning) tends to converge more quickly than RNSD. In Section 5, we show that this situation is similar for ill-posed problems, but that there is some unusual behavior in the convergence of BB that is not present in RNSD.

Example 4: If we take $M_0 = I$ and define $M_{k+1} = I - \frac{\mathbf{s}_k \mathbf{y}_k^T}{\mathbf{y}_k^T \mathbf{s}_k}$ where $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ and $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$ and choose $\alpha_k = \min_{\alpha>0} f(\mathbf{x}_k + \alpha \mathbf{r}_k)$ then we get the CGLS iterates [7]. Since CGLS is the conjugate gradient algorithm performed on the normal equations and the conjugate gradient algorithm minimizes the A-norm of the error at each iteration, CGLS minimizes the residual at each iteration. CGLS is usually considered superior to the other methods because it converges much faster, however, for ill-posed problems, this also means that the noise tends to corrupt the iterates much more quickly than the other methods. Note that other quasi-Newton methods, such as the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method and the Davidon-Fletcher-Powell method (DFP), when applied to the quadratic minimization problem, also produce the conjugate gradient iterates [14].

3 Regularization

It was stated in Section 1 that because A is ill-conditioned, and because there is noise in $\mathbf{b}$, it is very difficult to compute accurate approximations of $\mathbf{x}$. To see why this is the case, suppose we naively assume the noise is small enough to ignore, and compute

$$\mathbf{x}_{\text{naive}} = A^{-1}\mathbf{b}\,.$$

Let $A = U\Sigma V^T$ be the singular value decomposition (SVD) of A, where U and V are $n \times n$ orthogonal matrices, and $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$. If we denote the columns of U and V as $\mathbf{u}_i$ and $\mathbf{v}_i$, respectively, then

$$\mathbf{x}_{\text{naive}} = V\Sigma^{-1}U^T\mathbf{b} = \sum_{i=1}^{n} \frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{v}_i = \sum_{i=1}^{n} \left(\xi_i + \frac{\eta_i}{\sigma_i}\right)\mathbf{v}_i\,, \tag{8}$$

where $\mathbf{x} = \sum_i \xi_i \mathbf{v}_i$ and $\mathbf{n} = \sum_i \eta_i \mathbf{u}_i$.

For a continuous (infinite dimensional) ill-posed problem, the *Picard condition* [20]

$$\sum_{i=1}^{\infty} \frac{|(\mathbf{b}, \mathbf{u}_i)|^2}{\sigma_i^2} < \infty \,,$$

where $(\mathbf{b}, \mathbf{u}_i)$ denotes inner product, must hold in order for a solution to exist. This implies that for the discrete problem that the coefficients $|\mathbf{u}_i^T \mathbf{b}|$ must decay, on average, faster that σ_i. However, for discrete ill-posed problems the discrete Picard condition is not necessarily met since for our problems the right hand side is contaminated with noise. In other words, the coefficients $|(\mathbf{b}, \mathbf{u}_i)|$ do not necessarily decay faster than the corresponding singular values. Discrete ill-posed problems have the property that the singular values decay to, and cluster at 0. Thus, division by small singular values, σ_i's, magnifies the corresponding noise components, η_i as seen in (8). In addition, most discrete ill-posed problems have the additional property that the singular vectors, $\mathbf{v}_i$, tend to oscillate more for smaller singular values. Thus, the naive solution is horribly corrupted by large oscillating components. These properties of ill-posed problems, as well as the above explanation using the SVD, are well known; see, for example, [8, 11, 21] for more details.

Regularization methods replace the original matrix by a better-conditioned but related one in order to diminish the effects of noise in the data and produce a *regularized solution* to the original problem. There are several types of regularization methods. Some well known methods are Tikhonov, truncated singular value decomposition (TSVD), and iterative regularization [21]. This paper focuses on iterative regularization.

A general *regularization* approach to compute an accurate approximation of $\mathbf{x}$ can be formulated as a modification of the naive solution (8) [11]; specifically,

$$\mathbf{x}_{\text{filt}} = \sum_{i=1}^{n} \phi_i \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i \,, \tag{9}$$

where the *filter factors*, ϕ_i, satisfy $\phi_i \approx 1$ for large σ_i, and $\phi_i \approx 0$ for small σ_i. That is, the large singular value components of the solution are reconstructed, while the components corresponding to the small singular values are filtered out. Different choices of filter factors lead to different methods. For example, the truncated SVD (TSVD) solution is given by (9), where

$$\phi_i = \begin{cases} 1 & \text{if } i \le r \\ 0 & \text{if } i > r \end{cases} .$$

If $\Sigma_r^\dagger = \text{diag}(\frac{1}{\sigma_1}, \ldots, \frac{1}{\sigma_r}, 0, \ldots, 0)$, then the TSVD solution can be written as

$$\mathbf{x}_{\text{tsvd}} = A_r^\dagger \mathbf{b} = V \Sigma_r^\dagger U^T \mathbf{b} = \sum_{i=1}^{r} \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i \,. \tag{10}$$

The truncation index, r, is called a *regularization parameter*, whose specific value is problem dependent. It is nontrivial to choose an "optimal" regularization parameter, though good values can be estimated using the discrepancy principle [21], the Picard condition [10], the L-curve [13], generalized cross validation (GCV) [9] or some other scheme [8, 11, 21].

The quasi-Newton methods discussed in the previous section belong to a class of iterative regularization methods [8, 21]. Such iterative algorithms exhibit a *semi-convergent* behavior with respect to the relative error, $||\mathbf{x}_k - \mathbf{x}||/||\mathbf{x}||$, where $\mathbf{x}_k$ is the approximate solution at the kth iteration. That is, the relative error begins to decrease and, after some "optimal" iteration, begins to rise.

By stopping the iteration when the error is low, we obtain a good (regularized) approximation of the solution. The iteration index (rather, its reciprocal $\frac{1}{k}$) plays the role of the regularization parameter. As with other regularization methods, it is a nontrivial matter to choose an "optimal" stopping iteration. If chosen too large, the corresponding solution will be too corrupted with noise, and it will be impossible to compute a good solution. If chosen too small, too much information about the solution is lost, and thus it is impossible to compute a good solution. For large image restoration problems, the discrepancy principle and the L-curve can be used efficiently to estimate the stopping iteration. GCV is not practical because there is a division by the trace of $I - A_r^\dagger$ which can only be approximated. The Picard condition is also not a feasible choice because the spectral coefficients are needed.

3.1 Quasi-Newton Filter Factors

Like many regularization methods, quasi-Newton methods can be thought of as filter methods, however, their filter factors are a bit complicated. From Section 2 we know that the quasi-Newton iteration is $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k M_k \mathbf{r}_k$. Substituting $A^T\mathbf{b} - A^T A\mathbf{x}_k$ for $\mathbf{r}_k$, we have

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k M_k (A^T\mathbf{b} - A^T A\mathbf{x}_k)\,. \tag{11}$$

Assuming, without loss of generality, $\mathbf{x}_0 = \mathbf{0}$, and using (11), we see that

$$\mathbf{x}_{k+1} = P_k(A^T A)A^T\mathbf{b}\,, \tag{12}$$

where the polynomial P_k is defined as

$$P_k(\lambda) = P_{k-1}(\lambda) + \alpha_k M_k(1 - \lambda P_{k-1}(\lambda))\,, \quad P_{-1}(\lambda) = 0\,. \tag{13}$$

Substituting $A = U\Sigma V^T$ into (12), we obtain

$$\mathbf{x}_{k+1} = \sum_{i=1}^{n} \sigma_i^2 P_k(\sigma_i^2)\frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{v}_i\,. \tag{14}$$

Comparing (14) to (9) we see that the quasi-Newton filter factors are

$$\sigma_i^2 P_k(\sigma_i^2). \tag{15}$$

One of the first approaches to be studied for its iterative regularization properties is the Landweber (LW) method [8, 11, 21]. We noted in Section 2 that the LW iteration is a quasi-Newton method with $\alpha_k M_k = \alpha I$.

Combining equations 13 and 15 with $\alpha_k M_k = \alpha I$, we see that the Landweber filter factors simplify to

$$1 - (1 - \alpha\sigma^2)^k. \tag{16}$$

Unfortunately, the RNSD, BB and CGLS filter factors do not simplify as nicely as the LW filter factors. Recall from Section 2 that RNSD and BB are quasi-Newton methods with $\alpha_k M_k = \alpha_k I$ where α_k is chosen to minimize the residual at each iteration for RNSD while α_k is the previous RNSD step length for BB. From equation (13) we can see that the filter factors for RNSD and BB are $\sigma_i^2 P_k(\sigma_i^2)$ where

$$P_k(\lambda) = P_{k-1}(\lambda) + \alpha_k(1 - \lambda P_{k-1}(\lambda))\,, \quad P_{-1}(\lambda) = 0\,. \tag{17}$$

The filter factors for CGLS are derived in the same way as the quasi-Newton filter factors (see [22, 11]). Specifically, they are $\sigma_i^2 P_k(\sigma_i^2)$ where the polynomial P_k is defined as

$$P_k(\lambda) = \left(1 - \alpha_k\lambda + \frac{\alpha_k\beta_{k-1}}{\alpha_{k-1}}\right) P_{k-1}(\lambda) - \frac{\alpha_k\beta_{k-1}}{\alpha_{k-1}} P_{k-2} + \alpha_k \,, \tag{18}$$

where $P_{-1}(\lambda) = 0$ and $P_0(\lambda) = \alpha_0$, $\alpha_k = ||\mathbf{r}_{k-1}||_2^2/||A\mathbf{d}_{k-1}||_2^2$, $\beta_k = ||\mathbf{r}_k||_2^2/||\mathbf{r}_{k-1}||_2^2$, $\mathbf{d}$ is the step direction, and $\mathbf{r}$ is the residual, $A^T(\mathbf{b} - A\mathbf{x})$.

From equation (16) it is easy to see that the LW filter factors are approximately 1 for large singular values and approximately 0 for small singular values, however, it is difficult to see analytically how the filter factors for RNSD, BB and CGLS compare since $P_k(\lambda)$ are defined recursively. We can, however, make such a comparison experimentally. This will be done in Section 5, but first we consider the important problem of preconditioning.

4 Preconditioning

Preconditioning is used with iterative methods to accelerate the rate of convergence; that is, to reduce the number of iterations needed to compute a good approximation of the solution. The standard approach to preconditioning, when solving $A\mathbf{x} = \mathbf{b}$, is to construct a matrix, P, such that the singular values of $P^{-1}A$ are clustered around a point which is bounded away from zero, such as 1. More singular values clustered around a point, as well as tighter clusters, implies faster convergence.

At each iteration of a preconditioned method, a linear system of the form $P\mathbf{z} = \mathbf{w}$ is solved. If P is a good approximation to a severely ill-conditioned A, then P will also be very ill-conditioned, and thus inaccuracies in the data will be highly amplified when $P\mathbf{z} = \mathbf{w}$ is solved in the first iteration. Further iterations do not improve the situation, and, in general, there is no hope of recovering a good approximation of the solution. Thus, the standard approach to preconditioning cannot be used when solving ill-posed problems.

An approach for preconditioning ill-posed problems, first proposed in [10], is to construct a matrix P that clusters the large singular values around one, but leaves the small singular values alone. We can argue why this approach can be effective by constructing an "ideal" TSVD preconditioner[1] as follows. Let $A = U\Sigma V^T$ be the SVD of A, where $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$. Define P_k as

$$P_k = U\Sigma_k V^T \,, \tag{19}$$

where $\Sigma_k = \text{diag}(\sigma_1, \ldots, \sigma_k, 1, \ldots, 1)$, and k is the truncation index for the TSVD solution (10). Then the preconditioned system has the form $P^{-1}A = V\Delta V^T$, where $\Delta = \text{diag}(1, \ldots, 1, \sigma_{k+1}, \ldots, \sigma_n)$. Thus the large singular values are perfectly clustered at one, and are well separated from the small singular values. Regularization properties of the iterative methods imply that it takes only one iteration to compute a regularized solution. Though it is not practical to compute an SVD to use as an ideal preconditioner, it is possible in some applications to use an approximate SVD, or spectral decomposition.

For spatially invariant image restoration problems involving $n \times n$ images, A is an $n^2 \times n^2$ block Toeplitz matrix with Toeplitz blocks (BTTB). Matrix vector multiplications with A can be done using $O(n^2 \log n)$ arithmetic operations using fast Fourier transforms (FFT). Preconditioning such structured matrices has been thoroughly investigated in the literature; see the survey paper [3] and

[1] By "ideal" we mean a preconditioner constructed from the SVD of A.

the references therein. Moreover, many of these approaches have been applied to image restoration [10].

Although a variety of approaches to preconditioning have been proposed, the most popular is to use circulant approximations. The approximation we use is

$$\min ||A - P||_F$$

where, for two dimensional image restoration problems, the minimization is done over all matrices that are block circulant with circulant blocks (BCCB). These preconditioners can be constructed efficiently; see [3] for more details.

An important and useful property of BCCB matrices is that they can be diagonalized efficiently using FFTs. That is, every BCCB matrix can be written as $P = \mathcal{F}^*\Lambda\mathcal{F}$, where Λ is a diagonal matrix containing the eigenvalues of P, $\mathcal{F}$ is the unitary (two dimensional) discrete Fourier transform matrix, and $\mathcal{F}^*$ is the complex conjugate transpose of $\mathcal{F}$ [6]. Thus, a BCCB approximation of A provides an approximate spectral decomposition, which can be used in an analogous manner as outlined above for the ideal SVD preconditioner. That is, we determine a reasonable truncation index, and construct the preconditioner

$$P = \mathcal{F}^*\Lambda_k\mathcal{F}\,.$$

In a realistic problem, we must use a *parameter-choice* method, such as the discrepancy principal, generalized cross validation (GCV), L-curve, or some other method [8, 11, 21]. In our computations we use GCV, such as would be done for TSVD regularization. That is, assuming A is $n^2 \times n^2$, we find the index k that minimizes

$$G(k) = \frac{\sum_{j=k+1}^{n^2} |\hat{b}_j|^2}{(n^2-k)^2}\,, \tag{20}$$

where $\hat{\mathbf{b}} = \mathcal{F}\mathbf{b}$.

We remark that for computational purposes, the matrix $\mathcal{F}$ does not need to be constructed explicitly, since multiplying a vector by $\mathcal{F}$ is equivalent to applying an FFT to that vector, and multiplication by $\mathcal{F}^*$ is equivalent to applying an inverse FFT. Moreover, the eigenvalues of the BCCB matrix can be obtained by computing an FFT of its first column [6]. For an $n \times n$ image (thus an $n^2 \times n^2$ matrix A), it requires only $O(n^2 \log n)$ arithmetic operations to compute an FFT.

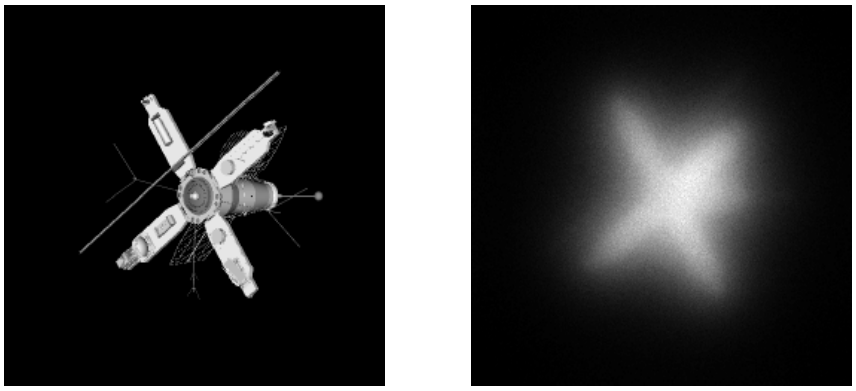
5 Numerical Experiments

In this section we observe the results from two different examples. The first is an image restoration test problem[2] that was developed at the US Air Force Phillips Laboratory, Lasers and Imaging Directorate, Kirtland Air Force Base, New Mexico. The image is a computer simulation of a field experiment showing a satellite as taken from a ground based telescope, and therefore represents an example of atmospheric blurring. The true and blurred, noisy images have 256×256 pixels, and are shown in Figure 1. The matrix A is an ill-conditioned $65,536 \times 65,536$ block Toeplitz matrix with Toeplitz blocks (BTTB). This data has been widely used in the literature for testing image restoration algorithms.

In the second example, we use the *enamel* test image (`enamel.tif`) from Matlab's Image Processing toolbox, artificially blur it using a cubic phase filter point spread function, and add .5% noise. The true and degraded images are shown in Figure 2.

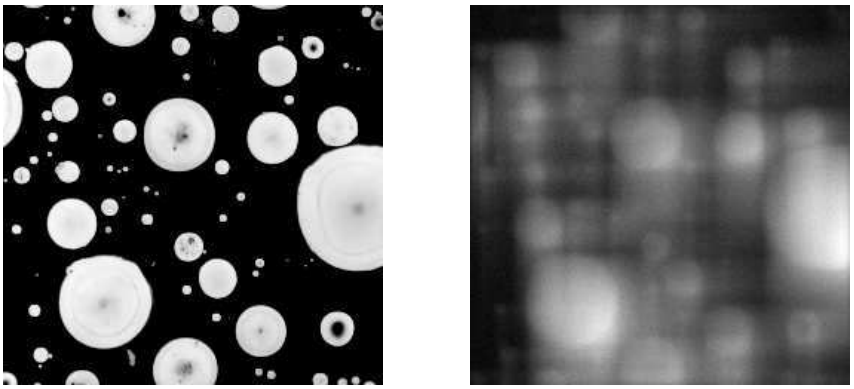
[2]This data is included in the image restoration package, *RestoreTools*, which can be obtained from http://www.mathcs.emory.edu/∼nagy/RestoreTools. Examples showing how to use the data in iterative methods may be found in an associated manuscript [16].

Our numerical tests were done using Matlab 6.1. Efficient implementations were done using *RestoreTools*, a Matlab package for image restoration [16].



(a) True image. (b) Blurred image.

Figure 1: `satellite` image data.



(a) True image. (b) Blurred image.

Figure 2: `enamel` image data.

In Figures 3 and 4, we show the filter factors at various iterations. The top row shows plots of the CGLS filter factors, the second row shows the BB filter factors, the third row shows the RNSD filter factors, and the bottom row shows the LW filter factors. Recall that we would like the filter factors to start off at 1, and eventually decay to 0. We observe that the filter factors for CGLS begin to oscillate, and this oscillation increases as the iterations proceed. This property of the CGLS filter factors was observed by Vogel [22]. In comparison, the filter factors for BB, RNSD and LW behave much more stably.

We also compared the relative error plots of the four methods on both `satellite` and `enamel` (see Figure 5). It is clear that RNSD and LW converge too slowly compared to CGLS and BB. BB doesn't converge as fast as CGLS, however the relative errors don't bounce back up as quickly as CGLS, making BB a nice alternative to CGLS when a good preconditioner is not available. Figure 6 compares the four preconditioned methods using the preconditioner described in Section 4. PLW (with $\omega = 0.05$) converges the slowest. PBB converges almost exactly the same as PRNSD. PBB

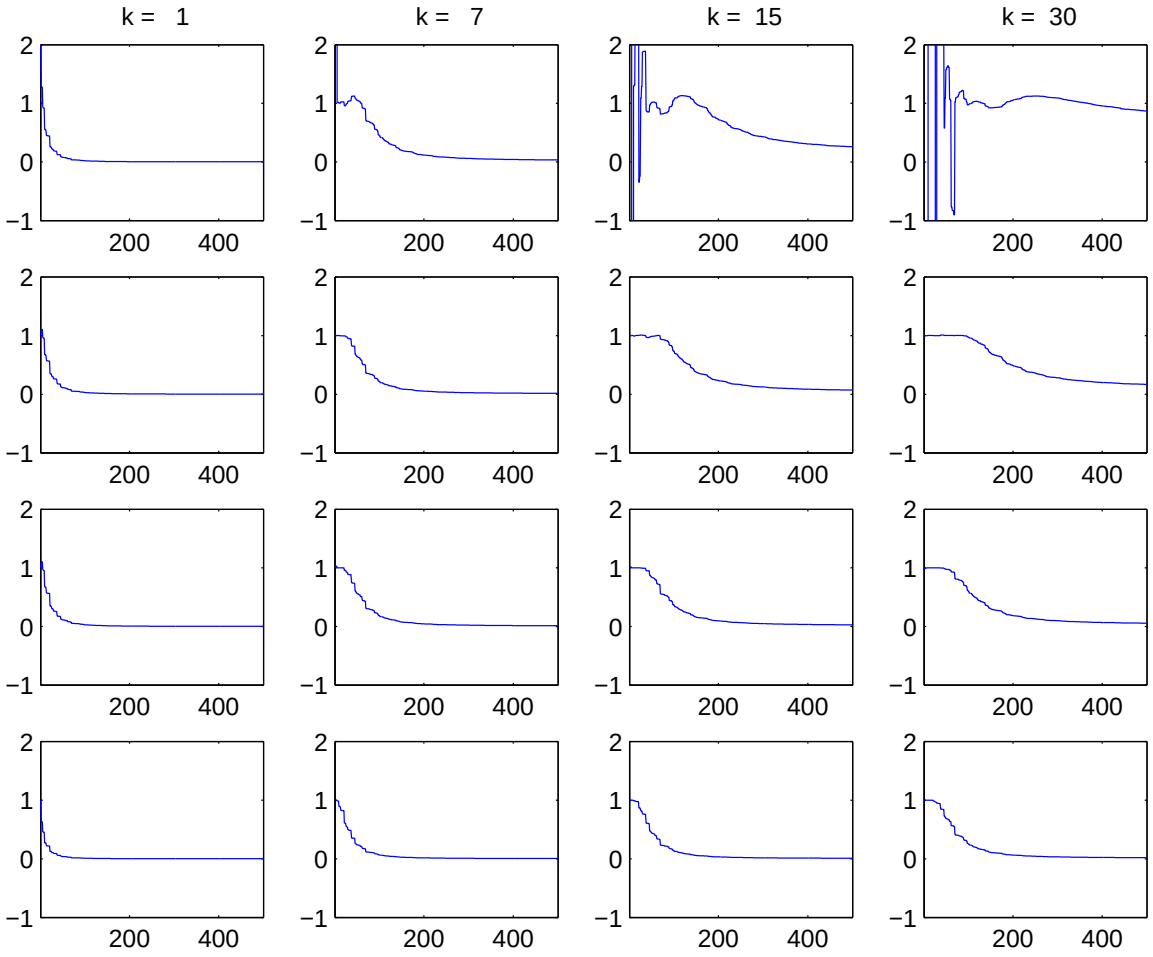


Figure 3: Filter factors for `satellite` at a few selected iterations of CGLS, BB, RNSD and LW.

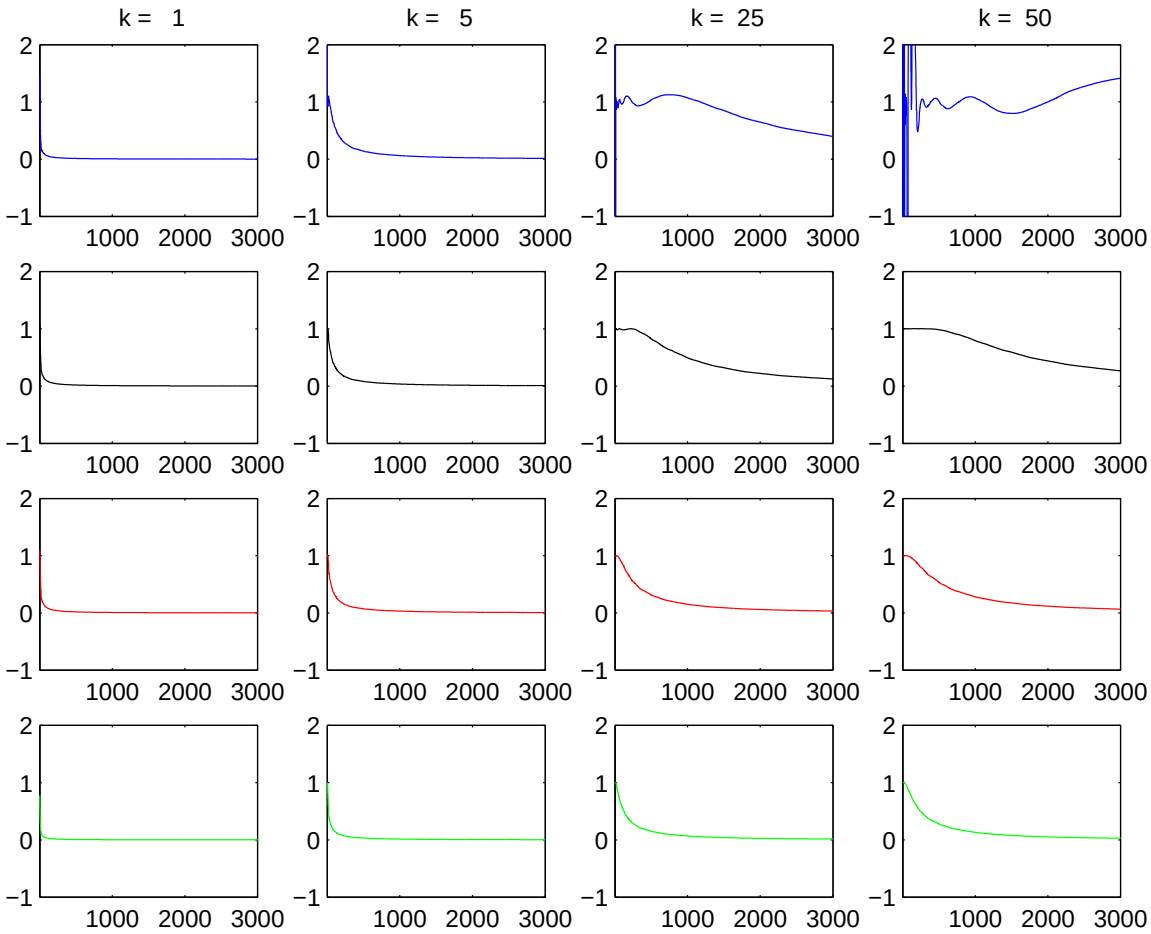
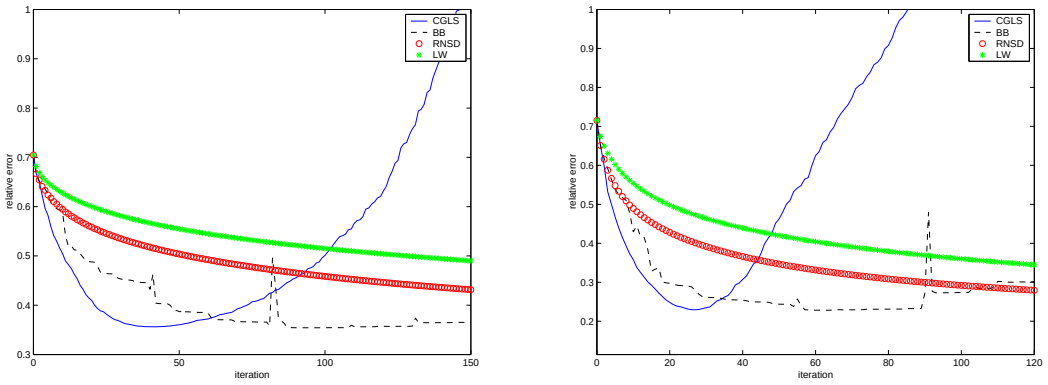


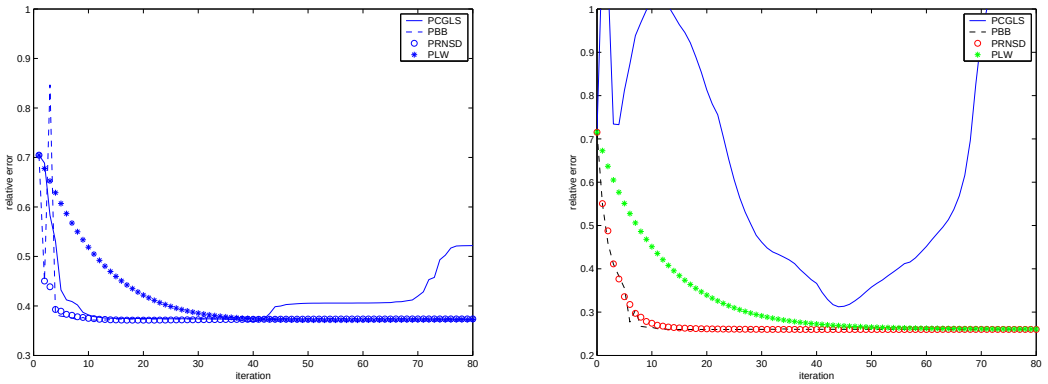
Figure 4: Filter factors for `enamel` at a few selected iterations of CGLS, BB, RNSD and LW.

jumps sharply after the first iteration on `satellite`, but then corrects itself. PCGLS converges then bounces back up for `satellite`. For the `enamel` test image, the results are similar, except that the convergence behavior for PCGLS is worse than using no preconditioning. The problem is that the preconditioner is too ill-conditioned for use in PCGLS. It may be possible to improve the convergence behavior by increasing the truncation index, but our experience is that if PCGLS exhibits such erratic convergence behavior, it is very difficult to find a good truncation index for the preconditioner. In such cases it is probably best to use no preconditioning with CGLS.



(a) `satellite`. (b) `enamel`.

Figure 5: Relative Error plots for CGLS, BB, RNSD, and LW



(a) `satellite`. (b) `enamel`.

Figure 6: Relative Error plots for PCGLS, PBB, PRNSD, and PLW

6 Concluding Remarks

In this paper we demonstrated that while CGLS is considered superior on well-posed problems due to its fast convergence, RNSD is a nice alternative when a good preconditioner is available and BB may be a nice alternative when a good preconditioner is not available. When we use PRNSD or BB, the relative error is reduced quickly in the early iterations, but it does not increase quickly after reaching an optimal level, as it does with CGLS. This kind of convergence behavior may

be preferable for applications where it is difficult to find an adequate stopping criteria. We also showed that the filter factors for BB, RNSD and LW are all much more stable than the filter factors for CGLS. The semi-convergent behavior for CGLS and PCGLS is much more prominent than for the other six methods implying a relationship between the stability of the filter factors and the semi-convergent behavior.

References

[1] J. Barzilai and J. Borwein, Two-point step size gradient methods, *IMA Journal of Numerical Analysis*, V. 8, pp. 141–148, 1988.

[2] T. F. Chan, An optimal circulant preconditioner for Toeplitz systems, *SIAM J. Sci. Stat. Comp.*, V. 9, pp. 766–771, 1988.

[3] R. H. Chan and M. K. Ng, Conjugate gradient methods for Toeplitz systems, *SIAM Review*, V. 38, pp. 427–482, 1996.

[4] T. F. Chan and J. A. Olkin, Preconditioners for Toeplitz-block matrices, *Numer. Algo.*, V. 6, pp. 89–101, 1993.

[5] W. Cheney and D. Kincaid, *Numerical Analysis, Second Edition*, Brooks/Cole Publishing Company, 1996.

[6] P. J. Davis, *Circulant Matrices*, Wiley, New York, 1979.

[7] J. E. Dennis, Jr. and J. J. Moré, Quasi-Newton methods, motivation and theory, *SIAM Review*, V. 19, No. 1, pp. 46–89, January 1977.

[8] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*, Kluwer Academic Publishers, Dordrecht, 2000.

[9] G. Golub, M. Heath and G. Whaba, Generalized cross-validation as a method for choosing a good ridge parameter, *Technometrics*, V.21, pp 215–223, 1979.

[10] M. Hanke, J. Nagy and R. Plemmons, Preconditioned iterative regularization, in *Numerical Linear Algebra*, de Gruyter, Berlin, L. Reichel, A. Ruttan and R. S. Varga, eds., pp. 141–163, 1993.

[11] P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems*, SIAM, Philadelphia, PA, 1997.

[12] P. C. Hansen. Regularization tools: A Matlab package for the analysis and solution of discrete ill-posed problems, *Numerical Algorithms*, V. 6, pp. 1–35, 1994.

[13] P. C. Hansen and D. P. O'Leary, The use of the L-curve in the regularization of discrete ill-posed problems, *SIAM J. Sci. Stat. Comput*, V. 14, pp. 1487–1503, 1993.

[14] T. G. Kolda, D. P. O'Leary, and L. Nazareth, BFGS with update skipping and varying memory, *SIAM J. Optim.*, V. 8, pp. 1060–1083, 1998.

[15] J. G. Nagy and K. M. Palmer, Steepest descent, CG, and iterative regularization of ill-posed problems, *BIT*, V. 43, pp. 1003–1017, 2004.

[16] J. G. Nagy, K. M. Palmer and L. C. Perrone, Iterative methods for image restoration: A Matlab object oriented approach, to appear in *Numerical Algorithms*, 2004.

[17] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, Boston, 1996.

[18] G. Strang, A proposal for Toeplitz matrix calculations, *Studies in Appl. Math.*, V. 74, pp. 171–176, 1986.

[19] H. J. Trussell, Convergence criteria for iterative restoration methods, *IEEE Trans. Acoust. Speech Signal Process.*, V. 31, pp. 129–136, 1983.

[20] J. M. Varah, A practical examination of some numerical methods for linear discrete ill-posed problems, *SIAM Review*, V. 21, pp. 100–111, January 1979.

[21] C. R. Vogel, *Computational Methods for Inverse Problems*. SIAM, Philadelphia, PA, 2002.

[22] C. R. Vogel, Solving ill-conditioned linear systems using the conjugate gradient method. Technical Report, Department of Mathematical Sciences, Montana State University, 1987.