Krylov Subspace Split Bregman Methods

Majed Alotaibi[a], Alessandro Buccini[b], Lothar Reichel[a]

[a] *Department of Mathematical Sciences, Kent State University, Kent, OH 44242, USA*
[b] *Dipartimento di Matematica e Informatica, Università di Cagliari, 09124 Cagliari, Italy*

Abstract

Split Bregman methods are popular iterative methods for the solution of large-scale minimization problems that arise in image restoration and basis pursuit. This paper investigates the possibility of projecting large-scale problems into a Krylov subspace of fairly small dimension and solving the minimization problem in the latter subspace by a split Bregman algorithm. We are concerned with the restoration of images that have been contaminated by blur and Gaussian or impulse noise. Computed examples illustrate that the projected split Bregman methods described are fast and give computed solutions of high quality.

Keywords: Split Bregman method, Krylov method, Golub-Kahan bidiagonalization, fixed point algorithm, cross validation

1. Introduction

This paper considers the solution of linear systems of equations

$$Au \approx f \tag{1.1}$$

where $A \in \mathbb{R}^{m\times n}$ is a large matrix and $f \in \mathbb{R}^m$ a given error-contaminated right-hand side. We are interested in computing a vector $u \in \mathbb{R}^n$ such that the left-hand side of (1.1) approximates f in a well-specified manner; see below. In our applications the matrix A is severely ill-conditioned and of ill-determined rank, i.e., its singular values decay to zero rapidly without a significant gap. The matrix A may be rank-deficient and we allow $m \geq n$ as well as $m < n$. This kind of matrices arises for instance in image restoration

Email addresses: malota15@kent.edu (Majed Alotaibi), alessandro.buccini@unica.it (Alessandro Buccini), reichel@math.kent.edu (Lothar Reichel)

problems. We will illustrate the performances of the methods described when applied to this kind of problems. Nevertheless, the methods described here also can be used in the context of other applications.

In our image restoration applications, the available image, which is represented by f, is contaminated by noise and blur. The latter is modeled by the matrix A. The data vector f can be written as

$$f = f_{\text{true}} + e, \tag{1.2}$$

where the vector $e \in \mathbb{R}^m$ is the unknown error in f; we will commonly refer to e as "noise." The vector f_{true} represents the unknown blur-contaminated, but noise-free vector, associated with f. In our computed examples, we let e model white Gaussian noise, random-valued impulse noise, or salt-and-pepper noise; see Section 5 for more details.

Let u_{true} denote the desired restoration of f. Thus, the entries of u_{true} are pixel values of the image ordered column-wise. Since the matrix A is severely ill-conditioned, the naive solution $A^\dagger f$, where $A^\dagger$ denotes the Moore-Penrose pseudo-inverse of A, usually gives a meaningless approximation of u_{true} due to propagation and amplification of the error e into the computed solution. An effective approach to deal with this difficulty is to replace (1.1) by a regularized problem, whose solution is less sensitive to the error in f. There are several common regularization methods; their choice should be informed by the statistical properties of the noise e and the available a-priori knowledge of u_{true}. When f is corrupted by Gaussian noise, we consider the minimization problem

$$\min_u \|Wu\|_1 + \frac{\mu}{2}\|Au - f\|_2^2, \tag{1.3}$$

and when f is contaminated by impulse noise, we instead solve the minimization problem

$$\min_u \|Wu\|_1 + \mu\|Au - f\|_1, \tag{1.4}$$

where $\|\cdot\|_2$ and $\|\cdot\|_1$ denote the Euclidean norm and the 1-norm, respectively, and $W \in \mathbb{R}^{r\times n}$, $r \geq n$, is a tight framelet analysis operator, i.e., it satisfies $W^T W = I$; see Section 6 for details. The parameter $\mu > 0$ in (1.3) and (1.4) is the so-called regularization parameter and controls the sensitivity of the problem to the noise. The choice of norms in the first term in the minimization problems (1.3) and (1.4) is dictated by the desire to determine a sparse vector Wu, as it is known that images u_{true} typically can be approximated well by a sparse representation in the framelet domain, i.e., many entries of the vector Wu_{true} vanish. We would like this also to

hold for the computed approximation of u_{true}. The choice of the norm of the second term in the minimization problems (1.3) and (1.4) can be justified by statistical considerations; see, e.g., [3, 7] for discussions.

The minimization problem (1.3) can be solved by the split Bregman algorithm [7, 9, 12], which is a modification of the iteration method described by Bregman [2]. The split Bregman method can be applied to the solution of the constrained minimization problem

$$\min_z \|z\|_1 \quad \text{such that} \quad AW^T z = f, \tag{1.5}$$

where $W \in \mathbb{R}^{r\times n}$, $r \geq n$, is a tight framelet analysis operator. Let z_* denote the solution of (1.5). One is then interested in determining $u_* = W^T z_*$; see [7].

This paper describes how the solution of large-scale problems of the forms (1.3) and (1.4) using split Bregman can be sped up by using appropriately chosen Krylov subspaces. Specifically, when we solve (1.3), we project the large matrix A onto a matrix of low rank by carrying out a few steps of Golub-Kahan bidiagonalization. This reduces the computational effort required to evaluate matrix-vector products. We illustrate this in Section 6. When we solve (1.4), we consider generalized Krylov subspaces; see, e.g., [20]. We would like to stress that the proposed method requires that the left inverse of W, i.e., a matrix $\widetilde{W}$ such that $\widetilde{W}W = I$, is easily computed. Therefore, the operator W cannot be chosen arbitrarily. Nevertheless, in our experience, framelet operators usually provide very accurate reconstructions of higher quality than differential operators, such as the total variation operator. Extension of the proposed methods to the case of general W will be subject of future research.

The choice of the regularization parameter μ in (1.3) and (1.4) is of vital importance, and an imprudent choice may lead to poor approximations of u_{true}. Two methods for determining the regularization parameter will be used in the computed examples: the fixed point method, described by Ito et al. [17], and cross validation, discussed, e.g., by Stone [21]. Both methods can be applied when no information about the norm of the error e is available. We choose these methods, because they can be applied both for Gaussian and impulse noise. However, we should mention that these methods are often referred to as "heuristic", because they may fail in certain situations, since they do not use the norm of the error vector e in (1.2). This is a consequence of the Bakushinskii veto [1]; see, e.g., [11, 18, 19] for discussions on heuristic methods.

Finally, we describe a two-step method for solving (1.4), which is well

suited for the situation when there is a large amount of impulse noise. This method is schematically similar to the method proposed by Cai et al. [8]. In the first step, we identify the location of (most of) the noise-contaminated pixels in the available image f, using the directional weighted median filter (DWMF) proposed by Dong and Xu [10], and remove these pixels from f. In the second step, we solve the minimization problem (1.4) using the split Bregman method with f replaced by this new reduced data vector.

This paper is organized as follows: Section 2 reviews the split Bregman method for the problems (1.3) and (1.4). In Section 3, we discuss the implementation of the projected split Bregman schemes using Krylov or generalized Krylov subspaces of fairly small dimensions, and Section 4 outlines the determination of the regularization parameter μ for the problems (1.3) and (1.4). In Section 5, we briefly review the two-step method that uses filtering for determining the regularization parameter. Computed examples are presented in Section 6, and concluding remarks can be found in Section 7.

2. Generalized Split Bregman Methods

We summarize the generalized split Bregman method proposed by Goldstein and Osher [12] in a more general setting than the one proposed above. Comments on how this method differs from the split Bregman scheme by Cai et al. [7] are also provided. Consider the minimization problem

$$\min_{u,d} \|d\|_1 + J(u) + \frac{\mu}{2} \left\| d - F(u) \right\|_2^2, \tag{2.1}$$

where $F : \mathbb{R}^n \to \mathbb{R}^r$ and $J : \mathbb{R}^n \to \mathbb{R}$ are non-linear and convex functions, respectively, $d \in \mathbb{R}^r$, $\mu > 0$ is a user-chosen parameter, and $\|F(u)\|_1$ is a convex functional. We are interested in the situations when $J(u) = \|Au - f\|_2^2$ or $J(u) = \|Au - f\|_1$, and $F(u) = Wu$.

Define the Bregman distance determined by a convex functional E at v as

$$D_E^p(u, v) = E(u) - E(v) - \langle p, u - v \rangle,$$

where $p \in \partial E(v)$, i.e., p is an element of the set of subgradients of E at v, and $\langle \cdot, \cdot \rangle$ stands for the standard inner product. The Bregman distance measures the "distance" between u and v in the sense that $D_E^p(u, v) \geq 0$ for all $u, v \in \mathbb{R}^n$, and $D_E^p(u, v) \geq D_E^p(w, v)$ for any $w \in \mathbb{R}^n$ on the line segment between u and v.

Let

$$E(u, d) = \|d\|_1 + J(u),$$

and let d^1 and u^1 be suitable initial vectors. The Bregman iteration applied to the solution of (2.1) iteratively computes, for $k = 1, 2, \dots$,

$$\begin{aligned}\left(u^{k+1}, d^{k+1}\right) &= \arg\min_{u,d} \; D_E^p(u, u^k, d, d^k) + \frac{\mu}{2}\left\|d - F(u)\right\|_2^2, \\ p_u^{k+1} &= p_u^k - \mu(\nabla F)^T\left(F\left(u^{k+1}\right) - d^{k+1}\right), \\ p_d^{k+1} &= p_d^k - \mu\left(d^{k+1} - F\left(u^{k+1}\right)\right),\end{aligned} \tag{2.2}$$

where

$$D_E^p(u, u^k, d, d^k) = E(u, d) - E(u^k, d^k) - \langle p_u^k, u - u^k\rangle - \langle p_d^k, d - d^k\rangle$$

and $(p_u^{k+1}, p_d^{k+1}) \in \partial E(u^{k+1}, d^{k+1})$. As shown by Yin et al. [22], when F is a linear operator, the iteration (2.2) simplifies to

$$\begin{aligned}\left(u^{k+1}, d^{k+1}\right) &= \arg\min_{u,d} \|d\|_1 + J(u) + \frac{\mu}{2}\left\|F(u) - d + b^k\right\|_2^2, \\ b^{k+1} &= b^k + \left(F\left(u^{k+1}\right) - d^{k+1}\right).\end{aligned} \tag{2.3}$$

Problem (2.3) can be solved by separately minimizing over u and d. This gives the following generalized split Bregman iteration described in [12] with appropriate initial vectors d^1 and b^1 and a fixed $M \geq 1$

$$\left\{\begin{array}{l} \text{for } \; k = 1, 2, \dots \\ \quad d^{k,1} = d^k, \\ \quad \left\{\begin{array}{l} \text{for } \; i = 1, 2, \dots, M, \\ \quad\quad u^{k,i} = \arg\min_u \; J(u) + \frac{\mu}{2}\left\|F(u) - d^{k,i} + b^k\right\|_2^2, \\ \quad\quad d^{k,i+1} = \arg\min_d \|d\|_1 + \frac{\mu}{2}\left\|d - F\left(u^{k,i}\right) - b^k\right\|_2^2, \\ \quad \text{end}, \end{array}\right. \\ \quad u^{k+1} = u^{k,M}, \\ \quad d^{k+1} = d^{k,M+1}, \\ \quad b^{k+1} = b^k + \left(F\left(u^{k+1}\right) - d^{k+1}\right), \\ \text{end}. \end{array}\right. \tag{2.4}$$

One may compute approximate solutions of higher quality by letting $M > 1$. This is illustrated in Section 6. Hence, the inner i-loop can be useful for

gaining higher accuracy in some applications. The vector $d^{k,i+1}$ in (2.4) is simply computed by applying the shrinkage operator. Thus,

$$d^{k,i+1} = \text{shrink}\left(F\left(u^{k,i}\right) + b^k, 1/\mu\right),$$

where

$$\text{shrink}\left(x, \alpha\right) = \text{sign}(x)\max\left(|x| - \alpha, 0\right), \tag{2.5}$$

and all operations in (2.5) are component-wise for any vector x.

2.1. The Split Bregman Method for Image Deblurring with Gaussian Noise

The minimization problem (1.3) is equivalent to

$$\min_{u,d} \|d\|_1 + \frac{\mu}{2}\|Au - f\|_2^2, \quad \text{with} \quad d = Wu. \tag{2.6}$$

An ℓ_2-norm penalty function is added to transform the constrained minimization problem (2.6) to the following unconstrained minimization problem

$$\min_{u,d} \|d\|_1 + \frac{\mu}{2}\|Au - f\|_2^2 + \frac{\lambda}{2}\left\|d - Wu\right\|_2^2, \tag{2.7}$$

where $\lambda > 0$ is a user-chosen parameter. However, the minimization problem (2.7) is not equivalent to (2.6) for any finite value of λ. One approach to determine an approximate solution of (2.6) is to solve (2.7) for a large value of λ, but numerical instability may arise. Therefore, Goldstein and Osher [12] suggested to use a moderate value of λ and transform the constraint minimization problem (2.6) into a sequence of unconstrained problems followed by a Bregman update.

Consider the unconstrained minimization problem (2.7). Since (2.7) is of the form (2.1), with $J(u) = \frac{\mu}{2}\|Au - f\|_2^2$ and $F(u) = Wu$, one can use the generalized split Bregman method discussed above. Hence, iteration (2.4) is applied to solve problem (2.7). This gives the split Bregman algorithm for solving (1.3) described by Algorithm 1.

When $M = 1$, a proof of convergence of the iterates u^k determined by Algorithm 1 to a solution of the minimization problem (1.3) is given in [7].

We can compute $u^{k,i}$ in Algorithm 1 by solving the linear system of equations

$$\left(\mu A^T A + \lambda I\right) u^{k,i} = \mu A^T f + \lambda W^T (d^{k,i} - b^k).$$

The vector $d^{k,i+1}$ in Algorithm 1 is computed by using the shrinkage operator

$$d^{k,i+1} = \text{shrink}\left(Wu^{k,i} + b^k, 1/\lambda\right).$$

Algorithm 1: SB-ℓ_2

Input: matrices $A \in \mathbb{R}^{m\times n}$ and $W \in \mathbb{R}^{r\times n}$, vector $f \in \mathbb{R}^m$, integer $M \geq 1$, positive constants λ and μ ;

Initialization: Let $u^1 = 0$ and $d^1 = b^1 = 0$;

for $k = 1, 2, ...$ **do**

  $d^{k,1} = d^k$;

  for $i = 1, 2, \ldots, M$ **do**

    $u^{k,i} = \arg\min_u \frac{\mu}{2}\|Au - f\|_2^2 + \frac{\lambda}{2}\left\|Wu - d^{k,i} + b^k\right\|_2^2$;

    $d^{k,i+1} = \arg\min_d \|d\|_1 + \frac{\lambda}{2}\left\|d - Wu^{k,i} - b^k\right\|_2^2$;

  end

  $u^{k+1} = u^{k,M}$;

  $d^{k+1} = d^{k,M+1}$;

  $b^{k+1} = b^k + \left(Wu^{k+1} - d^{k+1}\right)$;

end

Output: u^{k+1}

2.2. The Split Bregman Method for Image Deblurring with Impulse Noise

We borrow the splitting idea of the generalized split Bregman iteration to solve the minimization problem (1.4), where the ℓ_1-norm appears in both the fidelity and regularization terms. First, consider the following minimization problem, which is equivalent to (1.4),

$$\min_{u,d_1,d_2} \|d_1\|_1 + \mu\|d_2\|_1 \quad \text{s.t. } d_1 = Wu; \;\; d_2 = Au - f.$$

We then add two ℓ_2-penalty terms and consider the solution of the minimization problem

$$\min_{u,d_1,d_2} \|d_1\|_1 + \mu\|d_2\|_1 + \frac{\lambda}{2}\|d_2 - Au + f\|_2^2 + \frac{\lambda}{2}\|d_1 - Wu\|_2^2. \tag{2.8}$$

Letting

$$\begin{aligned} d &= \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}, \quad D = \begin{bmatrix} I & 0 \\ 0 & \mu I \end{bmatrix}, \\ K &= \begin{bmatrix} W \\ A \end{bmatrix}, \quad \tilde{f} = \begin{bmatrix} 0 \\ f \end{bmatrix}, \end{aligned} \tag{2.9}$$

we can express (2.8) as

$$\min_{u,d} \|Dd\|_1 + \frac{\lambda}{2} \left\| d - Ku + \tilde{f} \right\|_2^2. \tag{2.10}$$

This formulation is a special case of (2.1). Therefore, the minimization problem (2.10) can be solved by the generalized split Bregman iteration (2.4). This gives the split Bregman algorithm for solving the minimization problem (1.4) described by Algorithm 2.

If we set $M = 1$, the approach for showing convergence of the iterates u^k determined by Algorithm 2 to a solution of the minimization problem (1.4) can be shown by the technique used in [7].

Theorem 1. *Let u^* be a solution of the minimization problem (1.4). Assume that $\lambda, \mu > 0$, and let $M = 1$. Then the iterates u^k determined by Algorithm 2 satisfy*

$$\lim_{k\to\infty} \|Wu^k\|_1 + \mu\|Au^k - f\|_1 = \|Wu^*\|_1 + \mu\|Au^* - f\|_1.$$

Proof. The minimization problem

$$\min_u \left\| D\left(Ku - \tilde{f}\right) \right\|_1, \tag{2.11}$$

where D, K, and $\tilde{f}$ are defined in (2.9), is equivalent to the problem (1.4). Let u^* be a solution of (2.11). By the first order optimality conditions, u^* satisfies

$$0 = K^T Dp^*, \tag{2.12}$$

where $p^* \in \partial\|Dd^*\|_1$, and $d^* = Ku^* - \tilde{f}$. Let $b^* = \frac{1}{\lambda} Dp^*$. Using this property and (2.12), one can easily complete the proof by using the techniques in [7]. □

Letting

$$b^k = \begin{bmatrix} b_1^k \\ b_2^k \end{bmatrix}$$

and using the definitions (2.9), we compute $u^{k,i}$ in Algorithm 2 by solving the linear system of equations

$$\left(A^T A + I\right) u^{k,i} = A^T(f + d_2^{k,i} - b_2^k) + W^T(d_1^{k,i} - b_1^k). \tag{2.13}$$

Moreover, $d_1^{k,i+1}$ and $d_2^{k,i+1}$ are computed by

$$\begin{aligned} d_1^{k,i+1} &= \text{shrink}\left(Wu^{k,i} + b_1^k, 1/\lambda\right), \\ d_2^{k,i+1} &= \text{shrink}\left(Au^{k,i} - f + b_2^k, \mu/\lambda\right). \end{aligned}$$

Algorithm 2: SB-ℓ_1

Input: matrices $A \in \mathbb{R}^{m\times n}$ and $W \in \mathbb{R}^{r\times n}$, vector $f \in \mathbb{R}^m$, integer $M \geq 1$, positive constants λ and μ ;
Construct K, D, and $\tilde{f}$ defined in (2.9);
Let $u^1 = 0$ and $d^1 = b^1 = 0$;
for $k = 1, 2, ...$ **do**
  for $i = 1, 2, ..., M$ **do**
    $u^{k,i} = \arg\min_u \frac{\lambda}{2}\left\|Ku - \tilde{f} - d^{k,i} + b^k\right\|_2^2$;
    $d^{k,i+1} = \arg\min_d \|Dd\|_1 + \frac{\lambda}{2}\left\|d - Ku^{k,i} + \tilde{f} - b^k\right\|_2^2$;
  end
  $u^{k+1} = u^{k,M}$;
  $d^{k+1} = d^{k,M+1}$;
  $b^{k+1} = b^k + \left(Ku^{k+1} - \tilde{f} - d^{k+1}\right)$;
end
Output: u^{k+1}

3. Projected Split Bregman iteration using Krylov subspaces

This section discusses, in order, the situations when the data vector f is contaminated by Gaussian noise and by impulse noise.

3.1. Gaussian Noise Case

We propose a new solution method for the minimization problem (1.3) that projects the problem into a Krylov subspace of fairly small dimension. Specifically, we compute an approximate solution to the minimization problem (1.3) in the Krylov subspace

$$\mathbb{K}_\ell\left(A^TA, A^Tf\right) = \text{span}\left\{A^Tf, A^TAA^Tf, \left(A^TA\right)^2A^Tf, \ldots, \left(A^TA\right)^{\ell-1}A^Tf\right\}. \tag{3.1}$$

This is achieved by first determining an orthonormal basis for this space by applying $\ell \ll m$ steps of Golub–Kahan bidiagonalization to the matrix A with initial vector f; see [13]. This gives the decompositions

$$AV_\ell = U_{\ell+1}B_{\ell+1,\ell}, \quad A^T U_{\ell+1} = V_\ell B_{\ell,\ell}^T, \tag{3.2}$$

where the matrices $U_{\ell+1} = [u_1, u_2, \ldots, u_{\ell+1}] \in \mathbb{R}^{m\times(\ell+1)}$ and $V_\ell = [v_1, v_2, \ldots, v_\ell] \in \mathbb{R}^{n\times\ell}$ have orthonormal columns. The first column of $U_{\ell+1}$ is $f/\|f\|_2$ and the first column of V_ℓ is $A^T f / \left\|A^T f\right\|_2$. The matrices $B_{\ell+1,\ell} \in \mathbb{R}^{(\ell+1)\times\ell}$ and $B_{\ell,\ell} \in \mathbb{R}^{\ell\times\ell}$ are lower bidiagonal, and the matrix $B_{\ell,\ell}$ is the leading $\ell \times \ell$ submatrix of

$$B_{\ell+1,\ell} = \begin{bmatrix} \alpha_1 & & & & \\ \beta_2 & \alpha_2 & & & \\ & \beta_3 & \alpha_3 & & \\ & & \ddots & \ddots & \\ & & & \beta_\ell & \alpha_\ell \\ & & & & \beta_{\ell+1} \end{bmatrix},$$

where all entries α_j and β_j are positive. Moreover, the columns of V_ℓ span the Krylov subspace (3.1).

We seek an approximate solution of the minimization problem (1.3) in the range of V_ℓ as follows

$$\begin{aligned} & \min_{u\in\mathcal{K}_\ell\left(A^TA,A^Tf\right)} \|Wu\|_1 + \frac{\mu}{2}\|Au-f\|_2^2 \\ = & \min_{y\in\mathbb{R}^\ell} \|WV_\ell y\|_1 + \frac{\mu}{2}\|AV_\ell y - f\|_2^2 \\ = & \min_{y\in\mathbb{R}^\ell} \|WV_\ell y\|_1 + \frac{\mu}{2}\left\|B_{\ell+1,\ell}y - \|f\|_2 e_1\right\|_2^2, \end{aligned} \tag{3.3}$$

where $e_1 = [1, 0, \ldots, 0]^T$ denotes the first axis vector. The above derivation uses the decomposition (3.2), the fact that the columns of $U_{\ell+1}$ are orthonormal, and that $U_{\ell+1}^T f = \|f\|e_1$. The attraction of seeking a solution in the range of V_ℓ is that the large matrix A is replaced by the small matrix $B_{\ell+1,\ell}$.

We turn to the solution of the projected minimization problem (3.3), which is analogous to the minimization problem (1.3). We therefore can solve (3.3) by the SB-ℓ_2 method described in Algorithm 1. This results in

the following iterative method

$$\left\{\begin{array}{l} \text{for } \ k = 1, 2, \ldots \\ d^{k,1} = d^k, \\ \left\{\begin{array}{l} \text{for } \ i = 1, 2, \ldots, M, \\ y^{k,i} = \arg\min\limits_{y \in \mathbb{R}^\ell} \ \frac{\mu}{2} \left\| B_{\ell+1,\ell} y - \|f\|_2 e_1 \right\|_2^2 + \frac{\lambda}{2} \left\| W V_\ell y - d^{k,i} + b^k \right\|_2^2, \\ d^{k,i+1} = \arg\min\limits_{d} \|d\|_1 + \frac{\lambda}{2} \left\| d - W V_\ell y^{k,i} - b^k \right\|_2^2, \\ \text{end}, \end{array}\right. \\ y^{k+1} = y^{k,M}, \\ d^{k+1} = d^{k,M+1}, \\ b^{k+1} = b^k + \left(W V_\ell y^{k+1} - d^{k+1} \right). \end{array}\right. \tag{3.4}$$

At each iteration (3.4), we have

$$u^{k+1} = V_\ell y^{k+1}.$$

The details of the computations of $y^{k,i}$ and $d^{k,i+1}$ in (3.4) are similar to their counterparts in the SB-ℓ_2 method in Section 2.1. The summary of these computations is presented in Algorithm 3. We will refer to this method as the Split Bregman via Golub–Kahan method, in brief the SB-GK method.

We point out that for the computation of $y^{k,i}$ in Algorithm 3, we solve the least-squares problem

$$\min_y \left\| \left[\begin{array}{c} \mu^{1/2} B_{\ell+1,\ell} \\ \lambda^{1/2} I \end{array} \right] y - \left[\begin{array}{c} \mu^{1/2} \|f\|_2 e_1 \\ \lambda^{1/2} V_\ell^T W^T (d^{k,i} - b^k) \end{array} \right] \right\|_2^2$$

rather than solving the normal equations for improved numerical stability.

3.2. Impulse Noise Case

We now present our scheme for solving (1.4) by split Bregman iteration, where we reduce the dimension of the problem by projecting it into a low-dimensional Krylov-type subspace. However, instead of generating the solution subspace first, as done in the SB-GK method, we construct an appropriate space during the iterations. We start with a vector v_1, apply a k-iteration of SB-ℓ_1 described in Algorithm 2 in the one-dimensional subspace span$\{v_1\}$, and then increase the dimension of the solution subspace by adding a properly chosen vector v_2. At the second iteration, we look for

Algorithm 3: SB-GK

Input: matrices $A \in \mathbb{R}^{m\times n}$ and $W \in \mathbb{R}^{r\times n}$, vector $f \in \mathbb{R}^m$, integers $\ell, M \geq 1$, positive constants λ and μ ;

Let $d^1 = b^1 = 0$, $y^0 = y^1 = 0$ and $k = 1$;

Compute the decomposition (3.2) and store V_ℓ and $B_{\ell+1,\ell}$;

while $\left\|y^{k-1} - y^k\right\|_2 > tol \left\|y^{k-1}\right\|_2$ **do**

  $d^{k,1} = d^k$;

  for $i = 1, 2, ..., M$ **do**

    Solve for $y^{k,i}$,
    $\left(\mu B_{\ell+1,\ell}^T B_{\ell+1,\ell} + \lambda I\right) y^{k,i} = \mu B_{\ell+1,\ell}^T \|f\|_2 e_1 + \lambda V_\ell^T W^T (d^{k,i} - b^k)$;

    $d^{k,i+1} = \text{shrink}\left(WV_\ell y^{k,i} + b^k, 1/\lambda\right)$;

  end

  $y^{k+1} = y^{k,M}$;

  $d^{k+1} = d^{k,M+1}$;

  $b^{k+1} = b^k + \left(WV_\ell y^{k+1} - d^{k+1}\right)$;

  $k = k + 1$;

end

$u^{k+1} = V_\ell y^{k+1}$;

Output: u^{k+1}

a solution in span$\{v_1, v_2\}$. We iterate this way and increase the dimension of the search space by one after each iteration. In detail, we define the initial vector $v_1 = A^T f$ and set $V_1 = v_1 / \|v_1\|_2$. At iteration k of SB-ℓ_1, the columns of $V_k = [v_1, v_2, \ldots, v_k]$ form an orthonormal basis for the search subspace. Compute the QR factorization

$$AV_k = Q_A R_A, \tag{3.5}$$

where the matrix $Q_A \in \mathbb{R}^{m\times k}$ has orthonormal columns and the matrix $R_A \in \mathbb{R}^{k\times k}$ is upper triangular. We compute the vector $y^{k,i}$ by solving the minimization problem

$$\begin{aligned}
&\min_{u\in\mathcal{K}_k} \frac{\lambda}{2}\left\|Au - f - d_2^{k,i} + b_2^k\right\|_2^2 + \frac{\lambda}{2}\left\|Wu - d_1^{k,i} + b_1^k\right\|_2^2 \\
&= \min_{y\in\mathbb{R}^k} \frac{\lambda}{2}\left\|AV_k y - f - d_2^{k,i} + b_2^k\right\|_2^2 + \frac{\lambda}{2}\left\|WV_k y - d_1^{k,i} + b_1^k\right\|_2^2 \\
&= \min_{y\in\mathbb{R}^k} \frac{\lambda}{2}\left\|Q_A R_A y - f - d_2^{k,i} + b_2^k\right\|_2^2 + \frac{\lambda}{2}\left\|WV_k y - d_1^{k,i} + b_1^k\right\|_2^2.
\end{aligned}$$

The solution $y^{k,i}$ of the last equation satisfies the normal equations

$$\left(R_A^T R_A + I\right) y^{k,i} = R_A^T Q_A^T (f + d_2^{k,i} - b_2^k) + V_k^T W^T (d_1^{k,i} - b_1^k) \tag{3.6}$$

with a small $k \times k$ matrix. We compute the solution $y^{k,i}$ by solving the least squares problem

$$\min_{y\in\mathbb{R}^k} \left\| \begin{bmatrix} R_A \\ I \end{bmatrix} y - \begin{bmatrix} Q_A^T(f + d_2^{k,i} - b_2^k) \\ V_k^T W^T (d_1^{k,i} - b_1^k) \end{bmatrix} \right\|_2^2,$$

since the condition number of the matrix in the latter problem is the square root of the condition number of the system matrix in (3.6). The vector u^{k+1} can be determined by $u^{k+1} = V_k y^{k+1}$.

In Algorithm 2 the vector u^{k+1} does not need to be computed explicitly, since one can use y^{k+1} to compute the vectors d_1^{k+1}, b_1^{k+1}, d_2^{k+1}, and b_2^{k+1}, because the matrices AV_k and WV_k are stored. We compute the residual vector associated with the normal equations (3.6) by

$$r^{(k+1)} = A^T\left(AV_k y^{k+1} - f - d_2^{k+1} + b_2^{k+1}\right) + W^T\left(WV_k y^{k+1} + b_1^{k+1} - d_1^{k+1}\right). \tag{3.7}$$

This vector is the gradient for the unprojected minimization problem associated with (3.6). The solution subspace $\mathcal{R}(V_k)$, where $\mathcal{R}(M)$ denotes the

range of M, is expanded by including the vector $r^{(k+1)}/\|r^{(k+1)}\|_2$, i.e., we let

$$V_{k+1} = \left[V_k, \frac{r^{(k+1)}}{\|r^{(k+1)}\|_2} \right].$$

A reorthogonalization of $r^{(k+1)}$ against $\mathcal{R}(V_k)$ may be necessary to reduce the propagation of round-off errors. We refer to $\mathcal{R}(V_{k+1})$ as a generalized Krylov subspace; see [20]. One could expand the subspace $\mathcal{R}(V_k)$ by some other vector instead. However, numerical experiments show that the vector (3.7) is a good choice.

We remark that the QR factorization in (3.5) does not have to be computed from scratch for every k. Instead, it is updated according to

$$\begin{aligned} v_{\text{new}} &= \frac{r^{(k+1)}}{\|r^{(k+1)}\|_2}, \\ AV_{k+1} = [AV_k, Av_{\text{new}}] &= [Q_A, \tilde{q}_A] \begin{bmatrix} R_A & r_A \\ 0^T & \tau_A \end{bmatrix}, \end{aligned}$$

where

$$\begin{aligned} r_A &= Q_A^T (Av_{\text{new}}), \quad & q_A &= Av_{\text{new}} - Q_A r_A, \\ \tau_A &= \|q_A\|_2, & \tilde{q}_A &= q_A/\tau_A. \end{aligned}$$

We will refer to the method described as the Split Bregman via generalized Krylov subspace method, briefly as the SB-GKS method. Algorithm 4 summarizes the computations.

4. The Choice of Regularization Parameter

This section describes the methods we use to determine the regularization parameter μ for (1.3) and (1.4) in the computed examples. We outline two methods: the Fixed Point method and Cross Validation.

4.1. The Fixed Point method

This section describes the Fixed Point (FP) method proposed by Ito et al. [17] applied to the solution of (1.3). Let

$$\mathcal{J}_\mu(u) := \frac{\mu}{2}\|Au - f\|_2^2 + \|Wu\|_1,$$

where μ is the regularization parameter. Note that in this subsection we follow the notation of [17]. Therefore, μ here is the reciprocal to the μ in

Algorithm 4: SB-GKS

Input: matrices $A \in \mathbb{R}^{m\times n}$ and $W \in \mathbb{R}^{r\times n}$, vector $f \in \mathbb{R}^m$, integer $M \geq 1$, positive constants λ and μ ;

Let $d_1^1 = b_1^1 = 0$, $d_2^1 = b_2^1 = 0$, $y^0 = y^1 = 0$ and $k = 1$;

Compute $v = A^T f$ and $V_1 = v/\left\|v\right\|_2$;

Compute the QR factorization $AV_1 = Q_A R_A$;

while $\left\|y^{k-1} - y^k\right\|_2 / > tol \left\|y^{k-1}\right\|_2$ **do**

  $d_1^{k,1} = d_1^k$;

  $d_2^{k,1} = d_2^k$;

  for $i = 1, 2, ..., M$ **do**

    Solve for $y^{k,i}$

    $\left(R_A^T R_A + I\right) y^{k,i} = R_A^T Q_A^T (f + d_2^{k,i} - b_2^k) + V_k^T W^T (d_1^{k,i} - b_1^k)$;

    $d_1^{k,i+1} = \text{shrink}\left(WV_k y^{k,i} + b_1^k, 1/\lambda\right)$;

    $d_2^{k,i+1} = \text{shrink}\left(AV_k y^{k,i} + b_2^k - f, \mu/\lambda\right)$;

  end

  $y^{k+1} = y^{k,M}$;

  $d_1^{k+1} = d_1^{k,M+1}$;

  $d_2^{k+1} = d_2^{k,M+1}$;

  $b_1^{k+1} = b_1^k + \left(WV_k y^{k+1} - d_1^{k+1}\right)$;

  $b_2^{k+1} = b_2^k + \left(AV_k y^{k+1} - f - d_2^{k+1}\right)$;

  $r = A^T \left(AV_k y^{k+1} - f - d_2^{k+1} + b_2^{k+1}\right) + W^T \left(WV_k y^{k+1} - d_1^{k+1} + b_1^{k+1}\right)$;

  $V_{k+1} = \left[V_k, \frac{r}{\|r\|_2}\right]$; $AV_{k+1} = \left[AV_k, A\frac{r}{\|r\|_2}\right]$; $WV_{k+1} = \left[WV_k, W\frac{r}{\|r\|_2}\right]$;

  Update the QR factorization $AV_{k+1} = Q_A R_A$;

  $k = k + 1$;

end

$u^{k+1} = V_k y^{k+1}$;

Output: u^{k+1}

the rest of this paper. The choice μ in this section is due to the fact that the FP method is numerically more stable when implemented in this way.

The parameter μ is determined by minimizing the function

$$\Phi_\gamma(\mu) = \frac{\gamma^\gamma}{(1+\gamma)^{1+\gamma}} \frac{\tilde{F}^{1+\gamma}(\mu)}{\mu}, \tag{4.1}$$

where $\gamma > 0$, and the function $\tilde{F}$ is defined by

$$\tilde{F}(\mu) = \inf_u \mathcal{J}_\mu(u).$$

The minimizer of (4.1) is determined using the simple Fixed Point algorithm with stopping criterion $|\mu^{j+1} - \mu^j|/|\mu^j| \leq tol$, for a user defined tolerance

$$\begin{cases} \text{Choose } \mu^1 > 0, \\ \text{For } j = 1, 2, \ldots, \\ \quad \text{Solve } u^{j+1} = \arg\min_u \frac{\mu^j}{2}\|Au - f\|_2^2 + \|Wu\|_1, \\ \quad \text{Update } \mu^{j+1} = \frac{1}{\gamma}\frac{\|Wu^{j+1}\|_1}{\frac{1}{2}\|Au^{j+1}-f\|_2^2}, \\ \text{end.} \end{cases} \tag{4.2}$$

Ito et al. [17] provide a proof of local linear convergence of the above algorithm, however, as pointed out in [17], the algorithm may fail to converge. Moreover, the FP method is a heuristic rule that may fail to determine a suitable value of the regularization parameter also when the method converges. Nevertheless, in our experiments the FP method typically determines a suitable value of the regularization parameter.

A good choice of γ gives a regularization parameter μ that is close to the optimal one. The parameter γ depends weakly on the noise level. Thus it can take a wide range of values. There is no known criterion for choosing γ. The authors in [17] proposed a two-step procedure for choosing γ, but the method is not reliable. We, therefore, fix the parameter in our numerical examples.

We propose to implement the FP algorithm as follows. At each iteration j, the minimization problem in (4.2) is approximately solved by the SB-GK algorithm. We then compute μ^{j+1} using the approximate solution u^{j+1}. This process is repeated until the FP algorithm converges. We will refer to this scheme as the SB-GK-FP method. It is summarized in Algorithm 5.

4.2. Cross Validation

This section discusses how the Cross Validation (CV) method can be applied for determining a suitable value of the regularization parameter μ in

Algorithm 5: SB-GK-FP

Input: matrices $A \in \mathbb{R}^{m\times n}$ and $W \in \mathbb{R}^{r\times n}$, vector $f \in \mathbb{R}^m$, integers $\ell, M \geq 1$, positive constants γ, λ and μ ;
Let $d^1 = b^1 = 0$ and $\mu^1 = \mu$;
Compute the decomposition (3.2) and store V_ℓ and $B_{\ell+1,\ell}$;
while $|\mu^{j+1} - \mu^j|/|\mu^j| > tol1$ **do**
  $[u^{j+1}, y^{j+1}] = \text{SB-GK}\left(B_{\ell+1,\ell}, W, f, \mu^j, M, \lambda, V_\ell\right)$;
  $\mu^{j+1} = \frac{1}{\gamma}\frac{\|Wu^{j+1}\|_1}{\frac{1}{2}\|B_{\ell+1,\ell}y^{j+1} - \|f\|_2 e_1\|_2^2}$;
end
Output: u^{j+1}

the minimization problem (1.4). This method has previously been applied successfully to determine the regularization parameter for other minimization problems with application to the restoration of images that are contaminated by blur and impulse noise; see, e.g., [5]. The CV method splits the data f into two complementary sets: the training set and the testing set. Assume for notational simplicity that the testing set is made up of the first $d \ll m$ elements of $f \in \mathbb{R}^m$. The training set is then defined as

$$\widetilde{f} = [f_{d+1}, f_{d+2}, \cdots, f_m]^T \in \mathbb{R}^{m-d}.$$

Let $\widetilde{A} \in \mathbb{R}^{(m-d)\times n}$ be obtained by removing the first d rows from the matrix A. The training set is used for solving the minimization problem (1.4) with different regularization parameter values, and the testing set is used to validate these values. In actual computations, the elements of the training set are chosen randomly. Therefore, we propose to implement the CV method as follows. Consider solving (1.4) with some regularization parameter. Let I denote a set of d distinct random integers between 1 and n, and let $\widetilde{A}$ and $\widetilde{f}$ be defined by removing the rows with indices in the set I from A and f, respectively. Let $\{\mu_j\}_{j=1}^{\ell}$ be a set of positive regularization parameters, and let u_{μ_j} be given by

$$u_{\mu_j} := \arg\min_{u\in\mathbb{R}^n} \left\{ \|Wu\|_1 + \mu_j \|\widetilde{A}u - \widetilde{f}\|_1 \right\}.$$

Define the quantities

$$r_j = \sqrt{\sum_{i=1}^{d} \left(\left(Au_{\mu_j}\right)_i - f_i \right)^2}, \qquad 1 \leq j \leq \ell.$$

We identify as a suitable regularization parameter

$$\mu_{\widehat{j}}, \quad \text{with} \quad \widehat{j} = \arg\min_j r_j.$$

Let $\widehat{K}$ be a small positive integer. We carry out the above computations for $\widehat{K}$ different partitionings and average the regularization parameter value determined by each partitioning. At step $1 \leq k \leq \widehat{K}$, we consider a randomly selected set of d entries of the vector f as testing data, while the other $m - d$ entries are used as training data. This is known as the $\widehat{K}$-fold CV technique; it results in a less biased model, because it ensures that each entry in f may both appear in a training set and in a testing set. Each step provides a regularization parameter $\mu^{(k)}$ for $k = 1, 2, \ldots, \widehat{K}$ denoted by $\mu^{(k)}$. The chosen regularization parameter μ^* is the average of the parameter values $\mu^{(k)}$, i.e.,

$$\mu^* = \frac{1}{\widehat{K}} \sum_{k=1}^{\widehat{K}} \mu^{(k)}.$$

Finally, we use μ^* to compute u^* as

$$u^* := \arg\min_u \|Wu\|_1 + \mu^* \|Au - f\|_1.$$

This scheme is summarized in Algorithm 6, where we use Algorithm 4 to compute $u_{\mu_j}^{(k)}$. Let $u_{k,j}^{\text{init}} = \widetilde{A}^T \widetilde{f}$ for $j = 1, 2, \ldots, \ell$, $k = 1, 2, \ldots, \widehat{K}$, and let $v_{k,j}^{\text{init}} = \widetilde{A}^T \widetilde{f} / \left\| \widetilde{A}^T \widetilde{f} \right\|_2$ for $j = 1, 2, \ldots, \ell$, $k = 1, 2, \ldots, \widehat{K}$. These choices allow us to run the SB-GKS algorithm in parallel for different regularization parameter values.

5. Impulse Noise Filtering

The SB-GKS method is well suited for the restoration of images that have been contaminated by blur and impulse noise. We recall that f is said to be affected by random-valued impulse noise, if it has the entries

$$f_i = \begin{cases} a_i & \text{with probability } \sigma_p, \\ f_{\text{true}_i} & \text{with probability } 1 - \sigma_p, \end{cases} \tag{5.1}$$

where σ_p is referred to as the noise level and a_i is a number chosen uniformly at random in the interval $[a_{\min}, a_{\max}]$, which is the dynamic range of f_{true_i}. If $a_i \in \{a_{\min}, a_{\max}\}$, then the impulse noise is called salt-and-pepper noise.

Algorithm 6: SB-GKS-CV

Input: matrices $A \in \mathbb{R}^{m\times n}$ and $W \in \mathbb{R}^{r\times n}$, vector $f \in \mathbb{R}^m$, integers $d, \widehat{K}, M \geq 1$, positive constant λ, set of positive regularization parameters $\left\{\mu_j\right\}_{j=1}^{l}$;

for $k = 1, 2, \ldots \widehat{K}$ **do**

  Construct a set $I^{(k)}$ of d distinct random integers between 1 and n; let $\widetilde{A}$ and $\widetilde{f}$ be obtained by removing the rows with indices in $I^{(k)}$ from A and f, respectively;

  for $j = 1, 2, \ldots l$ **do**

    $u_{\mu_j}^{(k)} = \text{SB-GKS}\left(\widetilde{A}, W, \widetilde{f}, \mu_j, V_{k,j}^{\text{init}}\right)$;

    Compute $r_j^{(k)} = \sqrt{\sum_{i\in I^{(k)}} \left(\left(Au_{\mu_j}^{(k)}\right)_i - f_i\right)^2}$;

  end

  $j^* = \arg\min_{1\leq j\leq l}\left\{r_j^{(k)}\right\}$;

  $\mu^{(k)} = \mu_{j^*}$;

end

Compute $\mu^* = \frac{1}{\widehat{K}}\sum_{k=1}^{\widehat{K}} \mu^{(k)}$;

$u^* = \text{SB} - \text{GKS}(A, W, f, M, \lambda, \mu^*)$;

Output: u^*

The quality of the restorations computed by the SB-GKS method is not satisfactory when the noise level is high, e.g., when $\sigma_p \geq 40\%$. In this case, a median filter method can be used to detect some of the corrupted pixels in f and to replace them by the median value of their neighboring pixels. We will apply the directional weighted median filter (DWMF) proposed by Dong and Xu [10]. This filter detects a noisy pixel if the minimum value of the sum of weighted absolute differences between each pixel and its neighbors aligned in the four main directions exceeds a certain threshold. If the minimum of the sum of weighted absolute differences is larger than the threshold, the noisy pixel is replaced by a weighted median of the neighboring pixels values. The success of noise detections depends on how many neighboring pixels in the main four directions are not corrupted by noise. For a large value of σ_p, there is a high probability that the majority of the surrounding pixels are noisy. This can be deduced from the definition of the impulse noise (5.1), as the noisy pixels will be distributed uniformly in an image, with a high probability that some of them will be clustered in some neighborhood. Therefore, replacing the noisy pixel by a median value of its neighbors may produce blurred edges; see Figure 6.8.

We propose to use the DWMFs only for locating (some of) the noisy pixels. We then remove these pixels from our data f before applying the SB-GKS algorithm. Let J be a set of d distinct integers between 1 and n. These integers are the locations of the noisy pixels detected by the DWMF method. Let $\widetilde{A}$ and $\widetilde{f}$ be defined by removing the rows with index in J from A and f, respectively. We then apply the SB-GKS method using $\widetilde{A}$ and $\widetilde{f}$ as input to obtain the solution u, i.e.,

$$u := \arg\min_{u \in \mathbb{R}^n} \left\{ \|Wu\|_1 + \mu \|\widetilde{A}u - \widetilde{f}\|_1 \right\}.$$

We refer to this scheme as the filtered SB-GKS (F-SB-GKS) method.

6. Numerical Examples

We now consider some image deblurring examples to show the performances of the algorithms proposed in this paper. We refer the interested reader to [15] for more details on image deblurring; here we just recall that the matrix A is severely ill-conditioned, large, and structured, i.e., such that matrix-vector products with A can be performed cheaply using the `fft` algorithm; see, e.g., [15].

We consider two types of noise, Gaussian and impulse noise. In the first case the entries of e are independent realizations of a Gaussian random

variables with zero mean, and we refer to the ratio

$$\sigma_{\mathrm{g}} = \frac{\|e\|_2}{\|f_{\mathrm{true}}\|_2} \tag{6.1}$$

as the noise level. Impulse noise is defined in (5.1) and we refer to σ_p as noise level in this case. We express the noise levels σ_{g} and σ_p defined in (6.1) and (5.1), respectively, as percentages.

The dynamic range of $\hat{f}_i$ is $[0, 255]$ in our experiments, because each pixel is stored in 8 bits. Reflexive boundary conditions are imposed for all test images. We compare the SB-GK and SB-GK-FP methods with the projected linearized Bregman PLB method described in [4] and the SB-ℓ_2-CG and SB-ℓ_1-CG methods defined below. Results for the SB-GKS and F-SB-GKS methods for high impulse noise levels are presented. We compare the restored images in terms of the number of iterations, computing time, and the quality of the restored images measured by the Peak Signal-to-Noise Ratio (PSNR). The PSNR is defined by

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2}{m \sum_{i,j} \left(u_{i,j}^{\mathrm{res}} - u_{i,j}^{\mathrm{true}} \right)^2},$$

where $u_{i,j}^{\mathrm{res}}$ and $u_{i,j}^{\mathrm{true}}$ denote pixel values of the restored and original images at pixel (i, j), respectively.

The matrix A represents the blurring operator. For the construction of W, we use the linear B-spline tight frame system that is derived from a low-pass filter $W_0 \in \mathbb{R}^{n \times n}$ and two high-pass filters $W_1, W_2 \in \mathbb{R}^{n \times n}$. They are defined by

$$W_0 = \frac{1}{4} \begin{bmatrix} 3 & 1 & 0 & \dots & 0 \\ 1 & 2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 2 & 1 \\ 0 & \dots & 0 & 1 & 3 \end{bmatrix}, \quad W_1 = \frac{\sqrt{2}}{4} \begin{bmatrix} -1 & 1 & 0 & \dots & 0 \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ 0 & \dots & 0 & -1 & 1 \end{bmatrix},$$

and

$$W_2 = \frac{1}{4} \begin{bmatrix} 1 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ 0 & \dots & 0 & -1 & 1 \end{bmatrix}.$$

The corresponding regularization matrix W in two space-dimensions is given by

$$W = \begin{bmatrix} W_0 \otimes W_0 \\ W_0 \otimes W_1 \\ W_0 \otimes W_2 \\ W_1 \otimes W_0 \\ \vdots \\ W_2 \otimes W_2 \end{bmatrix},$$

where $\otimes$ denotes the Kronecker product. The regularization operator W is very sparse. This makes the evaluation of matrix-vector products with W and W^T inexpensive. In particular, the computational work of applying the framelet operators is linear in the number of unknowns. The matrix W satisfies $W^T W = I$. One may consider other function spaces, however, the good results achieved with the framelet basis are not guaranteed to carry over to other function spaces.

In Algorithms 2, 3, and 4, we set the regularization parameter μ to the value determined experimentally that yields a restored image that is closest to u^{true} in terms of PSNR value. The quality of the restored images is not very sensitive to the choice of λ. The value of λ only influences the rate of the convergence of the methods. We found empirically that setting $\lambda = 2$ results in (relatively) fast convergence. For the SB-GK method, we set $tol = 1 \times 10^{-4}$ in Algorithm 3. The dimension ℓ of the Krylov subspace in the SB-GK method needs not to be large. Using a Krylov subspace of larger dimension does not improve the accuracy. Here we set ℓ to a moderate value, $\ell = 11$.

We also compare the SB-GK and SB-GKS method to the split Bregman algorithm Algorithms 1 and 2, where we solve the linear system of equations (13) and (2.13) at each iteration by the Conjugate Gradient method. We implement the CGLS algorithm as described in [14] with stopping criterion 10^{-7} and let the initial guess be chosen as the most recently computed approximate solution u^k. We refer to this version of the split Bregman algorithm to as the SB-ℓ_2-CG method for the Gaussian noise case and as the SB-ℓ_1-CG method for the impulse noise case. We set the values of the regularization parameters in SB-ℓ_2-CG and SB-ℓ_1-CG to the same values of μ in SB-GK and SB-GKS, respectively. For the SB-ℓ_2-CG, SB-ℓ_1-CG, and SB-GKS methods, we set $tol = 5 \times 10^{-4}$ in Algorithms 1, 2, and 4. The iterations with CGLS in SB-ℓ_1-CG are terminated when the relative residual error is less than $tol = 1 \times 10^{-7}$. For a fair comparison with SB-GK, we set the maximum number of iterations of CGLS in SB-ℓ_2-CG to

11. Note that in most of our experiments the CGLS algorithm converges to the desired accuracy in less than 11 iterations. This implies that there is no need to precondition the system. For the SB-GK-FP method, we set $\gamma = 5$, $tol1 = 1 \times 10^{-3}$, and $\mu = 1$ in Algorithm 5. We set the parameters in Algorithm 6 for the SB-GKS-CV method to $\hat{K} = \ell = 8$, $d = \left[\frac{n}{200}\right]$, and the μ_j are 8 evenly spaced points between 10 and 90.

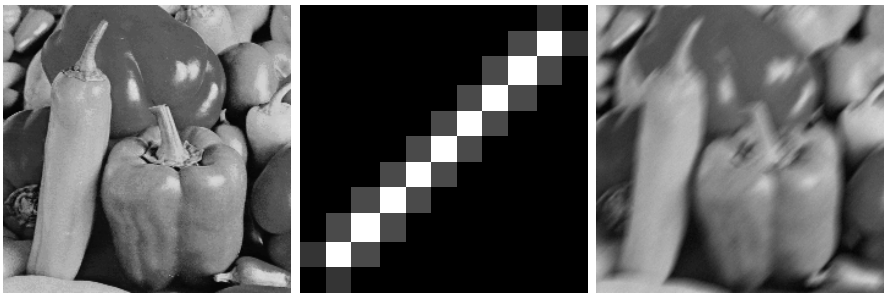
For the SB-GKS method, increasing the number of inner iterations, M, improves the quality of the computed solution. Also, there is a memory-time tradeoff when choosing M in the SB-GKS method, i.e., when we increase M, the memory requirement reduces due to a decrease of the number of outer iterations. However, the execution time tends to increase due to the cost of the evaluation of the shrinkage operator at each inner iteration; see Table 6.4. For the SB-GK and SB-ℓ_1-CG methods, M only plays a role in gaining extra accuracy in the restoration. Based on our numerical experience, increasing the number of inner iterations M to be larger than 3 does not give a significant improvement in the quality of the restored images. We set $M = 3$ in all algorithms.

The proposed methods are compared with the ℓ_p-ℓ_q adaptive and fixed minimization majorization methods referred to as AMM-GKSℓ_p-ℓ_q and FMM-GKSℓ_p-ℓ_q, respectively; see [16]. We let $p = 2$ and $q = \{0.1, 1\}$ for test images corrupted by Gaussian noise, and $p = 0.8$ and $q = 0.1$ as well as $p = 1$ and $q = 1$ for images corrupted by impulse noise. We set the maximum number of iterations for the AMM-GKS method to 100, and set the tolerance for both methods to 5×10^{-4}. Also, we compare the SB-GK-FP method to the modified cross-validation MM-GKS-MCV method and the generalized cross-validation MM-GKS-GCV method proposed in [5] and [6], respectively. We set $p = 2$ and $q = 0.1$ for both MM-GKS-MCV and MM-GKS-GCV methods. The SB-GKS-CV method is also compared to the smoothed version of the GCV i.e. GCV-Smooth method implemented in [5]. Note that, since the CV and MCV perform several runs of the MM-GKS algorithm, we will not report the number of iterations perform, but just the overall computational cost. All computations are carried out using MATLAB R2020b running on a Windows 10 laptop computer with a i5-8750H CPU @2.40 GHz and 6GB of RAM.

Pepper. Figure 6.1 shows an image with 236×236 pixels that is blurred by a PSF that simulates motion blur. White Gaussian noise is added to the blurred image with a noise level of 2%. The restorations achieved with the SB-ℓ_2-CG, PLB, SB-GK, and SB-GK-FP algoritms are shown in Figure 6.2. We report the PSNR values, number of iterations, and the CPU

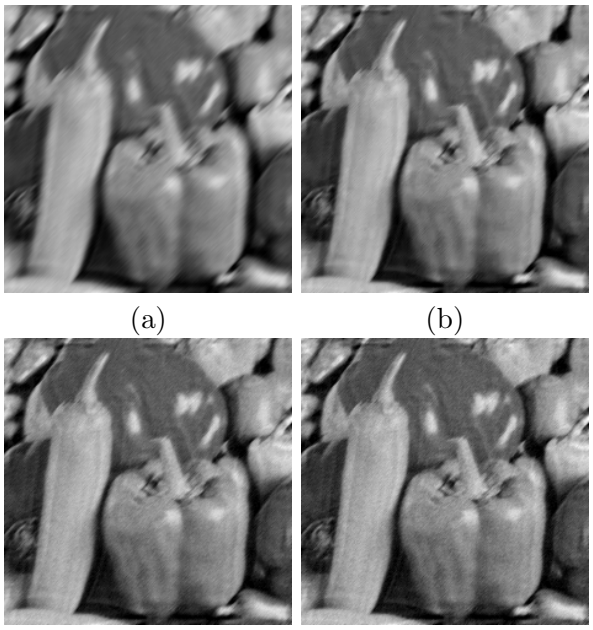
time measured in seconds in Table 6.1 for each method. All four methods yield very accurate restorations. We note that the image restored by the SB-GK method is slightly smoother than restorations determined by the PLB and SB-ℓ_2-CG method. The SB-GK-FP method determines a suitable value of μ and achieves a restored image of high quality. Overall, all methods required only a few iterations to converge and little CPU time. Also, we compare the restoration of the SB-GK method with the one determined by the SB-ℓ_2 method, where we use the generalized Krylov subspace V_k as a search space. The construction of V_k is similar to the technique used in the SB-GKS method. We refer to this method as SB-GK2. For this comparison, the pepper test image contains 1% white Gaussian noise. The restoration results are shown in Table 6.2. We notice that SB-GK is faster than SB-GK2 but the PSNR value of the image restored by SB-GK2 is higher. We point out that the dimension of the search space V_k of SB-GK2 is $k = 28$, which is larger than $\ell = 11$ in SB-GK. For the SB-ℓ_2-CG method, the PSNR value of the restored image is small compared to SB-GK and the total number of iterations and CPU time is higher in the SB-ℓ_2-CG method. But for both AMM-GKSℓ_2-$\ell_{0.1}$ and FMM-GKSℓ_2-$\ell_{0.1}$, we notice that the PSNR value is slightly higher but this small improvement in restoration quality is not enough to justify the much higher computational cost. SB-GK outperforms AMM-GKSℓ_2-ℓ_1 and FMM-GKSℓ_2-ℓ_1 in terms of both PSNR and CPU time. Both MM-GKS-MCV and MM-GKS-GCV determined a reasonable value of μ, but require a larger computational effort than the SB methods. We can also observe that the GCV-Smooth method produces a not very accurate reconstruction. This is due to the fact that the noise that corrupts the data is purely Gaussian and this algorithm is designed to handle impulse noise; see [6] for a discussion.

Goldhill. The image of size 246×246 pixels, shown in Figure 6.3, is blurred with a PSF that simulates average blur. We added 2% white Gaussian noise to the image. The restorations obtained with the SB-ℓ_2-CG, PLB, SB-GK, and SB-GK-FP methods are shown in Figure 6.4. The convergence plot is shown in Figure 6.12. We notice that the restoration determined by the SB-GK method displays more details of the original image compared to the restoration determined by the PLB method. The PSNR values, numbers of iterations, and execution times are reported in Table 6.1 for each method. Overall, we note that all methods converge quickly. We note that the SB-GKS-FP method determines a reasonable value of μ. The quality of the image restored by the SB-ℓ_2-CG method is not as high as the image restored by the SB-GK method. The computation time with SB-ℓ_2-CG is much



(a) (b) (c)

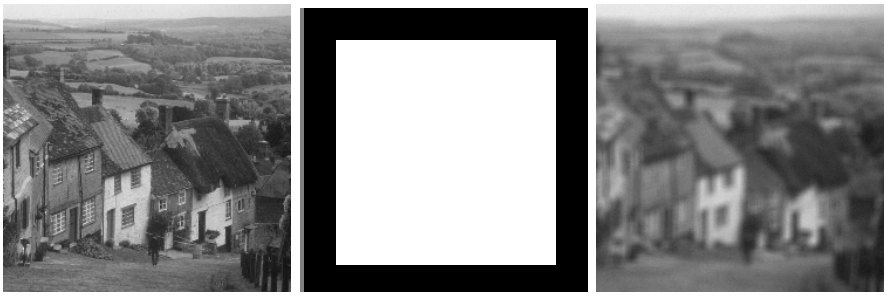
Figure 6.1: Peppers test case: (a) original pepper image (236×236 pixels), (b) PSF (11×11 pixels), (c) blurred image and noisy image (2% white Gaussian noise).



(a) (b)

(c) (d)

Figure 6.2: Pepper test problem restorations computed by: (a) SB-ℓ_2-CG (b) PLB (c) SB-GK (d) SB-GK-FP.



(a) (b) (c)

Figure 6.3: Goldhill test case: (a) original goldhill image (246×246 pixels), (b) PSF (9×9 pixels), (c) blurred image and noisy image (2% white Gaussian noise).

higher than with SB-GK. AMM-GKSℓ_2-$\ell_{0.1}$ and FMM-GKSℓ_2-$\ell_{0.1}$ is close to the proposed methods, but their computation requires more CPU time. AMM-GKSℓ_2-ℓ_1 and FMM-GKSℓ_2-ℓ_1 perform poorly compared to SB-GK in terms of the PSNR value and CPU time. Similarly to the previous example, the GCV-Smooth algorithm produces a reconstruction with a smaller PSNR value than the other considered methods.

Clock. An image of size 246×246 pixels is blurred by a PSF and is shown in Figure 6.5. We added 20% random-valued impulse noise to the image. The restorations determined by the SB-ℓ_1-CG, SB-GKS, and SB-GKS-CV methods are shown in Figure 6.6. The PSNR values, number of iterations, and the CPU time are reported in Table 6.1 for each method. We note that the SB-GKS method converged fairly quickly and determined a very accurate restoration. Also, the SB-GKS-CV method succeeded in determining an appropriate value of μ. Comparing the SB-GKS method with the SB-ℓ_1-CG method, we notice that the quality of the restored images obtained by the SB-GKS and SB-ℓ_1-CG methods is about the same. However, the SB-ℓ_1-CG method required more CPU time, because the CGLS method requires more matrix-vector product evaluations than the SB-GKS method. Similarly as above, the MM methods compute slightly more accurate reconstructions in some examples, however, the computational cost is extremely higher. We do not report results obtained with MM-GKS-MCV and MM-GKS-GCV as the methods did not produce accurate reconstructions.

We also compare SB-GKS to F-SB-GKS for the noise levels 30%, 40%, and 50%. From Table 6.3 and by visual inspection of Figure 6.7, we conclude that the F-SB-GKS method outperforms the SB-GKS method in terms of image quality. We also notice that the F-SB-GKS method requires more

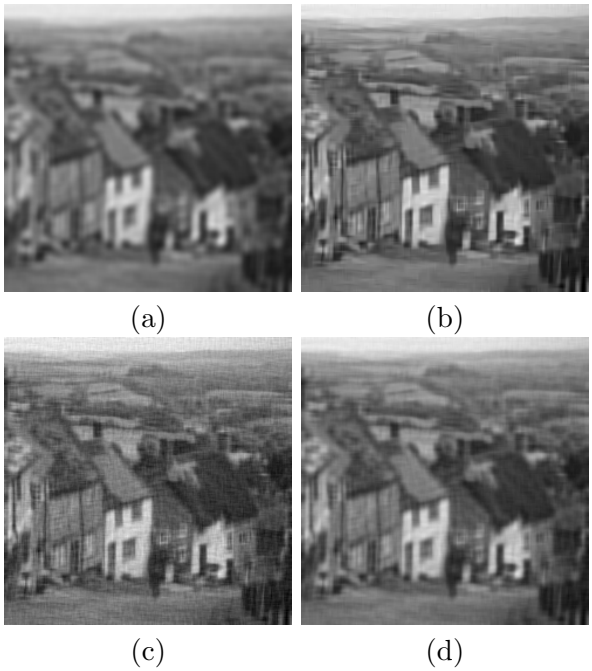
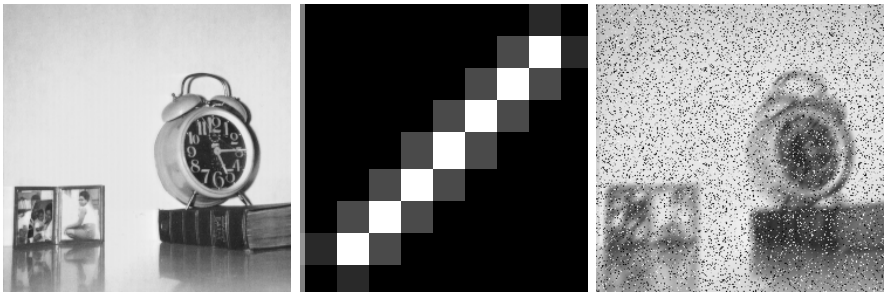


Figure 6.4: Goldhill test problem restorations computed by: (a) SB-ℓ_2-CG (b) PLB (c) SB-GK (d) SB-GK-FP.



(a) (b) (c)

Figure 6.5: Clock test case: (a) original clock image (246×246 pixels), (b) PSF (9×9 pixels), (c) blurred image and noisy image (20% random-valued impulse noise).

(a) (b) (c)

Figure 6.6: Clock test problem restorations computed by: (a) SB-ℓ_1-CG (b) SB-GKS (c) SB-GKS-CV

CPU time than the SB-GKS method, because the DWMF algorithm required several seconds to detect noisy pixels. We show the restoration of an example with a noise level of 50% using the DWMF method, where we replace the noisy pixels by median values of their neighboring pixels in Figure 6.8(a), and the restoration obtained by the DWMF method followed by the SB-GKS method in Figure 6.8(b). The DWMF method was able to correct most of the noisy pixels at the cost of introducing some blotches in the image as well as blurred edges. As a consequence, the image restored by the DWMF method followed by the SB-GKS method suffers from many defects. We can observe that the result obtained by GCV-Smooth is as accurate as the one computed by SB-GKS-CV. However, in SB-GKS-CV the algorithm runs 64 times, while the GCV-Smooth method runs only once. Since all the 64 runs of SB-GKS-CV are independent, the CPU time could be significantly reduced if the j-loop in Algorithm 6 is performed in parallel.

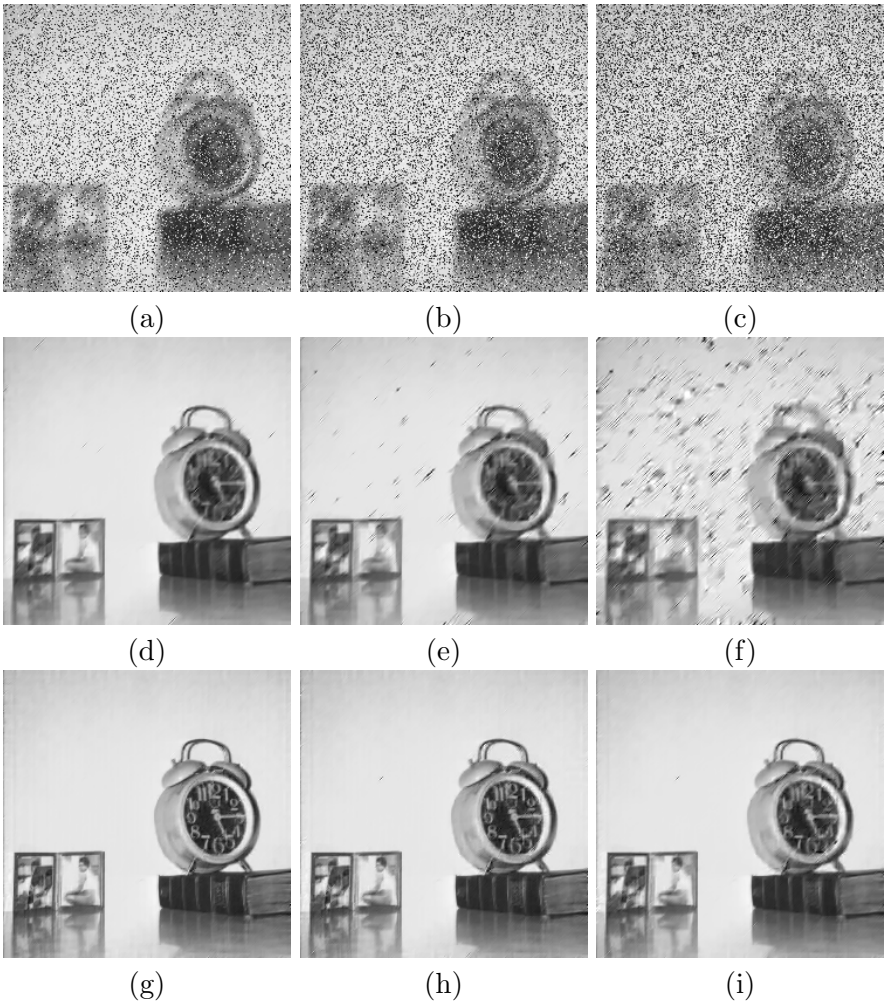
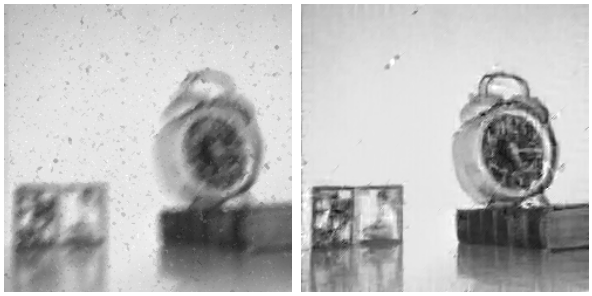


Figure 6.7: Clock test problem with different level of impulse noise. The first row reports the blurred and noisy image corrupted by random-valued impulse noise with a noise level of $30\%, 40\%$, and 50%, arranged from left to right respectively. The second row reports the reconstruction obtained with SB-GKS and the third row collects the ones computed by F-SB-GKS.



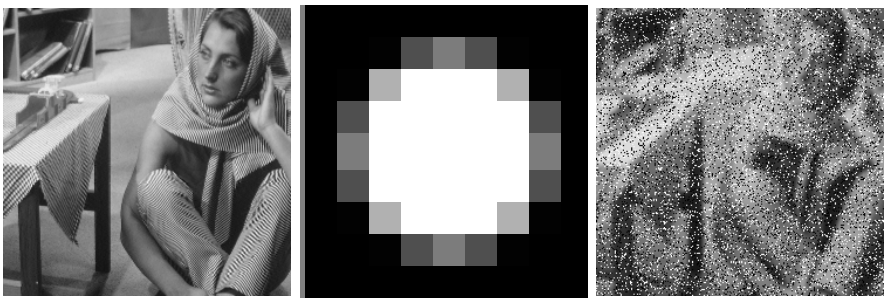
(a) (b)

Figure 6.8: Clock test problem with 50% noise level. The restorations computed by: (a) DWMF only (b) DWMF followed by SB-GKS

Barbara. An image of size 246×246 pixels is blurred by the PSF shown in Figure 6.9. We added 20% of salt-and-pepper noise to the image. The restored images obtained by the SB-ℓ_1-CG, SB-GKS, and SB-GKS-CV method show successful noise suppression; see Figure 6.10. The convergence plot is shown in Figure 6.12. We report the performances of the SB-GKS and SB-GKS-CV methods in terms of the execution time and PSNR-values in Table 6.1. The SB-GKS-CV method produced an image that is close in quality of the image determined by SB-GKS. The SB-GKS method outperformed SB-ℓ_1-CG in terms of the CPU time required until convergence. We obtain similar results as in the previous cases with AMM-GKS and FMM-GKS, i.e., they give slightly higher PSNR-values in some cases with a much higher computational cost. Moreover, as in the previous example, we do not display results for MM-GKS-MCV and MM-GKS-GCV as these methods did not produce meaningful reconstructions. Table 6.3 and Figure 6.11 illustrate the advantages of the two-step F-SB-GKS method for restoring images with high impulse noise levels. Finally, we stress that, in this case, the PSNR value of the image restored by GCV-Smooth is smaller than the one obtained by SB-GKS-CV.

7. Conclusions

This paper proposes two projected split Bregman schemes with the aid of Krylov subspaces for several image deblurring models. Methods for determining the regularization parameter are combined with some of these schemes and result in fully automatic and fast methods for image restoration. Moreover, these methods are faster than the AMM-GKS and FMM-GKS methods described in [16].



(a) (b) (c)

Figure 6.9: Barbara test case: (a) original Barbara image (246×246 pixels), (b) PSF (9×9 pixels), (c) blurred and noisy image (20% salt-and-pepper noise).

(a) (b) (c)

Figure 6.10: Barbara test problem restorations computed by: (a) SB-ℓ_1-CG (b) SB-GKS (c) SB-GKS-CV.

Figure 6.11: Barbara test problem with different level of impulse noise. The first row reports the blurred and noisy image corrupted by random-valued impulse noise with a noise level of $30\%, 40\%$, and 50%, arranged from left to right respectively. The second row reports the reconstruction obtained with SB-GKS and the third row collects the ones computed by F-SB-GKS.

Table 6.1: The PSNR value, number of iterations, and execution time in seconds for each example.

Example	Method	PSNR	Iterations	CPU time (seconds)
Pepper	SB-ℓ_2-CG	24.92	48	34.6
	PLB	25.9	60	3.95
	SB-GK	26.03	10	2.19
	SB-GK-FP	25.95	11	18.21
	AMM-GKSℓ_2-$\ell_{0.1}$	26.82	100	581.43
	AMM-GKSℓ_2-ℓ_1	25.65	100	455.2
	FMM-GKSℓ_2-$\ell_{0.1}$	26.64	143	181.15
	FMM-GKSℓ_2-ℓ_1	25.42	128	137.07
	MM-GKS-MCV	26.07	–	6.29×10^3
	MM-GKS-GCV	26.05	100	478.65
	GCV-Smooth	24.48	100	523.41
Goldhill	SB-ℓ_2-CG	24.86	71	55.3
	PLB	26.52	77	6.9
	SB-GK	26.55	9	1.95
	SB-GK-FP	25.97	15	21.76
	AMM-GKSℓ_2-$\ell_{0.1}$	26.98	100	643.57
	AMM-GKSℓ_2-ℓ_1	26.31	100	583.2
	FMM-GKSℓ_2-$\ell_{0.1}$	26.73	132	172.3
	FMM-GKSℓ_2-ℓ_1	26.02	102	131.9
	MM-GKS-MCV	23.32	–	6.41×10^3
	MM-GKS-GCV	26.08	100	650.86
	GCV-Smooth	25.19	100	689.12
Clock	SB-ℓ_1-CG	30.49	66	49.3
	SB-GKS	31.14	72	13.2
	SB-GKS-CV	29.9	–	735.5
	AMM-GKS$\ell_{0.8}$-$\ell_{0.1}$	31.79	100	668.07
	AMM-GKSℓ_1-ℓ_1	30.05	100	492.1
	FMM-GKS$\ell_{0.8}$-$\ell_{0.1}$	28.98	177	228.96
	FMM-GKSℓ_1-ℓ_1	27.57	143	193.09
	GCV-Smooth	29.96	100	636.49
Barbara	SB-ℓ_1-CG	24.78	101	69.28
	SB-GKS	24.84	91	14.84
	SB-GKS-CV	23.84	–	696
	AMM-GKS$\ell_{0.8}$-$\ell_{0.1}$	25.32	100	651.9
	AMM-GKSℓ_1-ℓ_1	24.36	100	452.55
	FMM-GKS$\ell_{0.8}$-$\ell_{0.1}$	25.17	331	578.34
	FMM-GKSℓ_1-ℓ_1	24.15	283	417.03
	GCV-Smooth	21.04	100	636.49

Table 6.2: The PSNR value, number of iterations, and execution time in seconds for the SB-GK and SB-GK2 algorithms in the Pepper test case. We do not report these results for the other methods as they are similar.

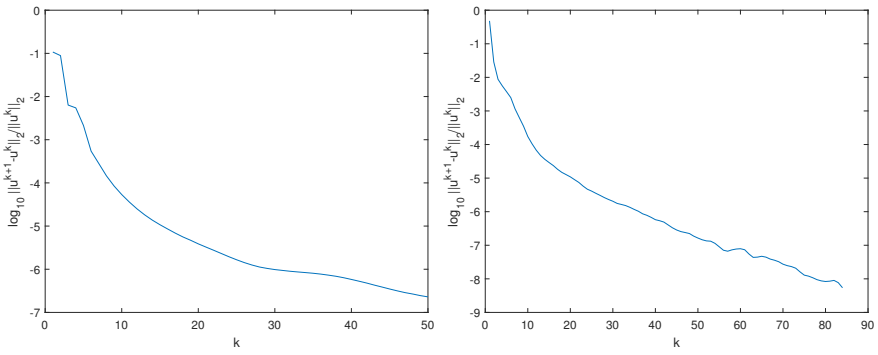
Method	PSNR	Iterations	CPU time (seconds)
SB-GK	26.16	8	2.22
SB-GK2	26.77	28	8.43

Table 6.3: The PSNR value, number of iterations, and execution time in seconds for SB-GKS and F-SB-GKS.

Example	Method	Noise Level	PSNR	Iterations	CPU time (seconds)
Clock	SB-GKS	30%	28.2	63	10
		40%	25.45	70	11.33
		50%	19.98	72	12.2
	F-SB-GKS	30%	33.05	68	15.73
		40%	31.24	69	15.97
		50%	29.24	59	14.60
Barbara	SB-GKS	30%	24.43	100	20.44
		40%	23.38	81	13.67
		50%	22.61	86	14.32
	F-SB-GKS	30%	25.31	121	26.55
		40%	24.8	102	26.08
		50%	22.94	101	22.7

Table 6.4: The PSNR value, number of iterations, and execution time in seconds for SB-GK and SB-GKS with different value of M

Example	Method	M	PSNR	Iterations	CPU time (seconds)
Goldhill	SB-GK	1	26.07	19	1.48
		2	26.40	14	1.62
		3	26.55	9	1.95
Barbara	SB-GKS	1	29.29	106	6.5
		2	30.32	76	8.67
		3	31.14	72	13.2



(a) (b)

Figure 6.12: The convergence plot of the restoration of: (a) Goldhill using SB-GK (b) Barbara using SB-GKS.

Acknowledgments

The authors would like to thank the referees for insightful comments that helped improve this manuscript. A.B. is a member of the GNCS group of INdAM and that partially founded his research with the project "Regularization Methods and Models for large scale inverse ill-posed problems".

References

[1] A. B. Bakushinskii, Remarks on the choice of regularization parameter from quasioptimality and relation tests, USSR Comput. Math. Math. Phys., 24 (1984), pp. 181–182.

[2] L. Bregman, The relaxation method of finding the common points of convex sets and its application to the solution of problems in convex optimization, USSR Comput. Math. Math. Phys., 7 (1967), pp. 200–217.

[3] A. Buccini, O. De la Cruz Cabrera, M. Donatelli, A. Martinelli, and L. Reichel, Large-scale regression with non-convex loss and penalty, Appl. Numer. Math., 157 (2020), pp. 590–601.

[4] A. Buccini, M. Pasha, and L. Reichel, Linearized Krylov subspace Bregman iteration with nonnegativity constraint, Numer. Algorithms, 87 (2021), pp. 1177–1200.

[5] A. Buccini and L. Reichel, An ℓ_p-ℓ_q minimization method with cross-validation for the restoration of impulse noise-contaminated images, J. Comput.Appl. Math., 375, (2020), pp. 112–824.

[6] A. Buccini and L. Reichel, Generalized Cross Validation for ℓ^p-ℓ^q minimization, Numer. Algorithms 88, (2021), pp. 1595–1616.

[7] J.-F. Cai, S. Osher, and Z. Shen, Split Bregman methods and frame based image restoration, Multiscale Model. Simul., 8 (2009), pp. 337–369.

[8] J.-F. Cai, R. Chan, and M. Nikolova, Fast two-phase image deblurring under impulse noise, J. Math. Imaging Vision, 36 (2010), pp. 46–53.

[9] V. De Simone, D. di Serafino, and M. Viola, A subspace-accelerated split Bregman method for sparse data recovery with joint ℓ_1-type regularizers, Electron. Trans. Numer. Anal., 53 (2020), pp. 406–425.

[10] Y. Dong and S. Xu, A new directional weighted median filter for removal of random-valued impulse noise, IEEE Signal Processing Letters, 14 (2007), pp. 193–196.

[11] H. W. Engl, M. Hanke, and A. Neubauer, Regularization of Inverse Problems, Kluwer, Dordrecht, 1996.

[12] T. Goldstein and S. Osher. The split Bregman algorithm for L1-regularized problems, SIAM J. Imaging Sci., 2, (2009), pp. 323–343.

[13] G. H. Golub and C. F. Van Loan, Matrix Computations, 4th ed. Johns Hopkins University Press, Baltimore, 2013.

[14] P. C. Hansen, Discrete Inverse Problems, SIAM, Philadelphia, 2010.

[15] P. C. Hansen, J. Nagy, and D. O'Leary, Deblurring Images: Matrices, Spectra, and Filtering, SIAM, Philadelphia, 2006.

[16] G. Huang, A. Lanza, S. Morigi, L. Reichel, and F. Sgallari, Majorization–minimization generalized Krylov subspace methods for $\ell_p-\ell_q$ optimization applied to image restoration, BIT. Numerical Mathematics, 57, (2017), pp. 351–378.

[17] K. Ito, B. Jin, and T. Takeuchi, A regularization parameter for nonsmooth Tikhonov regularization, SIAM J. Sci. Comput., 33 (2011), pp. 1415–1438.

[18] S. Kindermann, Discretization independent convergence rates for noise level-free parameter choice rules for the regularization of ill-conditioned problems, Electron. Trans. Numer. Anal., 38 (2011), pp. 233–257.

[19] S. Kindermann and K. Raik, A simplified L-curve method as error estimator, Electron. Trans. Numer. Anal., 53 (2020), pp. 217–238.

[20] J. Lampe, L. Reichel, and H. Voss, Large-scale Tikhonov regularization via reduction by orthogonal projection, Linear Algebra Appl., 436 (2012), pp. 2845–2865.

[21] M. Stone, Cross-validatory choice and assessment of statistical prediction, J. Royal Stat. Soc., series B, 36 (1977), pp. 111–147.

[22] W. Yin, S. Osher, D. Goldfarb, and J. Darron, Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing, SIAM J. Imaging Sci., 1 (2008), pp. 143–168.