Advances in Computational Mathematics manuscript No.
(will be inserted by the editor)

Limited memory restarted ℓ^p-ℓ^q minimization methods using generalized Krylov subspaces

Alessandro Buccini · Lothar Reichel

Received: date / Accepted: date

Abstract Regularization of certain linear discrete ill-posed problems, as well as of certain regression problems, can be formulated as large-scale, possibly nonconvex, minimization problems, whose objective function is the sum of the p^{th} power of the ℓ^p-norm of a fidelity term and the q^{th} power of the ℓ^q-norm of a regularization term, with $0 < p, q \leq 2$. We describe new restarted iterative solution methods that require less computer storage and execution time than the methods described by [Huang et al.,*Majorization-minimization generalized Krylov subspace methods for ℓ_p-ℓ_q optimization applied to image restoration.* BIT (2017)]. The reduction in computer storage and execution time is achieved by periodic restarts of the method. Computed examples illustrate that restarting does not reduce the quality of the computed solutions.

Keywords ℓ^p-ℓ^q minimization, inverse problem, regression, iterative method

Mathematics Subject Classification (2010) 65F10, 65R32, 90C26

1 Introduction

We are concerned with the solution of linear systems of equations of the form

$$A\mathbf{x} + \boldsymbol{\eta} = \mathbf{b}^{\delta}, \tag{1}$$

A. Buccini
Department of Mathematics and Computer Science
University of Cagliari
09124 Cagliari, Italy
E-mail: alessandro.buccini@unica.it

L. Reichel
Department of Mathematical Sciences
Kent State University
Kent, OH 44242, USA
E-mail: reichel@math.kent.edu

where the matrix $A \in \mathbb{R}^{m\times n}$ is ill-conditioned; the singular values of matrices of interest to us decay to zero with increasing index number with no significant gap between the small singular values, $\mathbf{x} \in \mathbb{R}^n$ is the unknown signal that we would like to recover, and $\mathbf{b}^\delta \in \mathbb{R}^m$ is an available data vector. The vector $\boldsymbol{\eta} \in \mathbb{R}^m$ represents unknown errors in the data vector. This vector may stem from measurement errors, faulty collection equipment, as well as discretization; it is often referred to as "noise". Linear systems of equations (1) of this kind are commonly referred to as *linear discrete ill-posed problems*. They arise, e.g., in image restoration and when discretizing linear ill-posed problems, such as Fredholm integral equations of the first kind with a smooth kernel; see, e.g., [13, 18] for discussions on ill-posed problems. Linear discrete ill-posed problems also arise in regression; see, e.g., [3].

Due to the ill-conditioning of A and the error $\boldsymbol{\eta}$ in $\mathbf{b}^\delta$, straightforward solution of $A\mathbf{x} \approx \mathbf{b}^\delta$, e.g., in the least-squares sense typically does not yield a useful approximation of the desired vector $\mathbf{x}$ in (1). We, therefore, will determine an approximation of this vector by solving an ℓ^p-ℓ^q minimization problem

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^n} \left\{ \frac{1}{p} \left\| A\mathbf{x} - \mathbf{b}^\delta \right\|_p^p + \frac{\mu}{q} \left\| L\mathbf{x} \right\|_q^q \right\}, \tag{2}$$

where $0 < p, q \leq 2$ and $\|\mathbf{x}\|_p^p = \sum_{j=1}^n |x_j|^p$. With slight abuse of notation, we will refer to $\|\mathbf{x}\|_p$ as the ℓ^p-norm of $\mathbf{x}$ for any $p > 0$, even though it is not a norm for $p < 1$. The first term in (2) is referred to as the *fidelity term* and the second term as the *regularization term*. The *regularization parameter* $\mu > 0$ balances the influence of these two terms on the solution $\mathbf{x}^*$ of (2). The matrix $L \in \mathbb{R}^{s\times n}$ often is referred to as a *regularization matrix*. We will assume that

$$\mathcal{N}(A) \cap \mathcal{N}(L) = \{\mathbf{0}\}, \tag{3}$$

where $\mathcal{N}(M)$ denotes the null space of M. Violation of this assumption trivially implies that the solution of (2) is not unique.

When $p = q = 2$, problem (2) is a Tikhonov regularization problem in general form; when $p \geq 1$ and $q \geq 1$, the problem is convex, but non-smooth if either $p = 1$ or $q = 1$. If $p < 1$ or $q < 1$, then the problem is non-convex and non-smooth.

We remark that the model (2) can be extended to allow more than one fidelity and regularization terms. The use of more than one fidelity term is straightforward and should be considered when this is likely to yield a solution $\mathbf{x}^*$ of higher quality than when only one fidelity term is considered. Application of two or more regularization terms has been discussed in the context of Tikhonov regularization ($p = q = 2$); see, e.g., [2, 5, 14, 16, 21]. This sometimes can result in computed solutions $\mathbf{x}^*$ of higher quality than when only one regularization terms is used. However, the computational effort required to determine suitable values of several regularization parameters is much larger than when determining only one parameter. Hence, use of Tikhonov regularization with more than one regularization term is attractive only if this results in an approximation of $\mathbf{x}_{\text{exact}}$ of significantly higher quality than when only one

regularization term is used. Similarly, the inclusion of more than one regularization term in the model (2) is only attractive when this results in a solution $\mathbf{x}^*$ of much higher quality than the model with only one regularization term. We, therefore, will not consider this extension of the model (2) in the present paper.

The choices of p and q in (2) is important, and briefly discuss how these parameters should be determined, starting with the parameter p. Let $\mathbf{b}$ denote the unknown noise-free data vector associated with the noise-contaminated vector $\mathbf{b}^\delta$, i.e., $\mathbf{b}^\delta = \mathbf{b} + \boldsymbol{\eta}$. Assume for the moment that $\mathbf{b} \in \text{range}(A)$. Then there is a vector $\mathbf{x}_0$ such that $A\mathbf{x}_0 - \mathbf{b}^\delta = \boldsymbol{\eta}$. This suggests that if $\boldsymbol{\eta}$ represents white Gaussian noise, then $p = 2$ is an appropriate choice. On the other hand, when $\boldsymbol{\eta}$ contains impulse noise (defined below), one would like to choose a "norm" that weighs outliers less than the Euclidean norm; one typically lets $0 < p \leq 1$.

We say that the data vector $\mathbf{b}^\delta = [(\mathbf{b}^\delta)_1, (\mathbf{b}^\delta)_2, \ldots, (\mathbf{b}^\delta)_m]^T$ is corrupted by impulse noise if

$$(\mathbf{b}^\delta)_j = \begin{cases} r_j & \text{with probability } \sigma, \\ (\mathbf{b})_j & \text{with probability } 1-\sigma, \end{cases} \qquad j = 1, 2, \ldots, m,$$

where $0 < \sigma < 1$ and r_j is the realization of a random variable with uniform distribution in the dynamic range of the entries of $\mathbf{b}$. In theory, $p = 0$ would be a good choice in the presence of impulse noise in $\mathbf{b}^\delta$, where the ℓ^0-norm counts the number of nonvanishing entries of a vector. However, the solution of minimization problems (2) with this norm is NP-hard and, therefore, not attractive to use in computations. It is popular to use the ℓ^1-norm instead of the ℓ^0-norm, because it secures convexity. However, ℓ^p-norms with $0 < p < 1$ provide better approximations of the ℓ^0-norm; see, e.g., [9] for an illustration. A Bayesian justification for choosing $0 < p < 1$ is presented in [3].

We turn to the choice of q. The value of q is informed by a-priori information about the sparsity of the desired solution $\mathbf{x}^*$ of (2). In imaging problems, the restored image usually has a sparse representation when expressed in wavelet or framelet bases, i.e., many of the coefficients in these representations vanish. In this case, letting $0 < q < 1$ usually provides satisfactory restorations; see, e.g., [8] for an illustration. Moreover, the discrete total variation operator applied to the desired solution often is sparse; see, e.g., [24] for illustrations.

A proof of the regularization properties of (2) is provided in [6], and a Bayesian derivation of the model is discussed in [3]. An application to supersaturated design is described in [4].

This paper presents two new methods for the solution of the minimization problem (2). These methods are based on the majorization-minimization (MM) methods described by Huang et al. [20] and Lanza et al. [24], but require less execution time and computer memory. The methods described in [20, 24] are iterative and are based on projecting the problem (2) into generalized Krylov subspaces of increasing dimension; the dimension is increased by one in each iteration. However, the MM methods in [20, 24] may require many

iterations to converge; computed examples reported in [20, 24] show that hundreds of iterations may be necessary to satisfy the stopping criterion for some examples. This large number of iterations results in a solution subspace of large dimension. For large-scale problems, the storage requirement for a basis for the solution subspace may be substantial. Moreover, QR factorizations of tall skinny matrices, with the number of columns equal to the dimension of the solution subspace, have to be computed or updated in every iteration. The computational effort required for these computations is significant when the dimension is large.

Extensive computational experience suggests that, when the dimension of the solution subspace is large, the last vectors added to the subspace often are not important for improving the quality of the computed solution. Therefore, our approach to reducing the memory requirement of the MM algorithms in [20] is to simply restart the algorithm periodically. We will describe and illustrate the performance of this approach.

Restarting iterative methods to reduce the memory requirement is not new. For instance, consider the GMRES method [28], which is a popular iterative method for the solution of boundary value problems for linear elliptic partial differential equations. These are *discrete well-posed problems*. Iterations with GMRES typically are restarted periodically to reduce the memory requirement and the computational effort per iteration. The number of iterations required to determine an approximate solution of satisfactory quality may be larger when using restarts than without restarting GMRES, because restarting affects the solution subspace used. However, restarting does not reduce the quality of the computed solution if sufficiently many iterations are performed.

Restarting iterative methods for the solution of discrete ill-posed problems is more delicate. The computed solution of (2) depends on the matrices A and L, the data vector $\mathbf{b}^\delta$, the regularization parameter $\mu > 0$, and the solution subspace chosen. The use of an unsuitable subspace can reduce the quality of the computed solution, independently of how many iterations are carried out. Restarting affects the solution subspace used and, therefore, may reduce the quality of the computed solution. The computed examples reported in this paper illustrate that our restarted methods yield computed solutions of the same quality as the nonrestarted methods described [20] with less computational effort.

This paper is organized as follows: Section 2 briefly reviews the MM algorithms described in [20] and Section 3 presents new memory- and CPU time-saving schemes and shows their convergence properties. We remark that the convergence proof for the MM algorithms in [20] does not carry over to the limited memory schemes of the present paper. However, the convergence proof of the present paper carries over to the methods in [20], and sheds light on their performance. A few numerical examples with application to image restoration are described in Section 4, and Section 5 contains concluding remarks.

The focus of this paper is to improve the algorithms described in [20], and we refer to [20, 24] for discussions of related work. Here we only mention the

papers by Chan and Liang [10] and Rodriguez and Wohlberg [27] that were an inspiration for the methods described in [20, 24].

2 Majorization-minimization in generalized Krylov subspaces

We review two iterative methods described in [20] for the minimization of (2). Each iteration consists of a majorization step and a minimization step. In the majorization step a smoothed ℓ^p-ℓ^q functional $\mathcal{J}_\varepsilon$, defined below, is majorized by a quadratic functional whose value and gradient agree with those of $\mathcal{J}_\varepsilon$ at the current approximation $\mathbf{x}^{(k)}$ of the sought minimum, which we denote by $\mathbf{x}_\varepsilon^*$. In the minimization step this majorant is minimized and its unique minimizer is the new approximation $\mathbf{x}^{(k+1)}$ of $\mathbf{x}_\varepsilon^*$. Two approaches to determine quadratic majorants are described in [20]. We will outline both.

Majorization step. Introduce the functional

$$\mathcal{J}(\mathbf{x}) = \frac{1}{p}\left\|A\mathbf{x}-\mathbf{b}^\delta\right\|_p^p + \frac{\mu}{q}\left\|L\mathbf{x}\right\|_q^q \tag{4}$$

associated with the minimization problem (2). We are primarily interested in the situation when $0 < \min\{p, q\} < 1$. Then the functional (4) is neither convex nor differentiable. The construction of the quadratic majorants requires the functional to be continuously differentiable. We therefore introduce the smoothed functional

$$\mathcal{J}_\varepsilon(\mathbf{x}) = \frac{1}{p}\sum_{j=1}^{m}\Phi_{p,\varepsilon}\left((A\mathbf{x}-\mathbf{b}^\delta)_j\right) + \frac{\mu}{q}\sum_{j=1}^{s}\Phi_{q,\varepsilon}\left((L\mathbf{x})_j\right)$$

for some small $\varepsilon > 0$, where

$$\Phi_{s,\varepsilon}(t) = \begin{cases} |t|^s & \text{for } s > 1, \\ \left(t^2+\varepsilon^2\right)^{s/2} & \text{for } 0 < s \leq 1, \end{cases} \tag{5}$$

is a differentiable function of t. It follows that $\mathcal{J}_\varepsilon(\mathbf{x})$ is everywhere differentiable. We will comment on the choice of ε in Section 4.

Our aim is to determine a solution $\mathbf{x}_\varepsilon^*$ of the problem

$$\min_{\mathbf{x}\in\mathbb{R}^n}\mathcal{J}_\varepsilon(\mathbf{x}). \tag{6}$$

When $\min\{p, q\} > 1$, the functional $\mathcal{J}_\varepsilon(\mathbf{x})$ is strictly convex and therefore has a unique minimum. However, when $0 < \min\{p, q\} < 1$, the functional (6) is not convex. The methods outlined seek to determine a local minimum or stationary point in the latter situation.

Let $\mathbf{x}^{(k)}$ be an available approximate solution of the problem (6). The majorant referred to as "adaptive" in [20] is determined by constructing a quadratic approximation of each component of $\mathcal{J}_\varepsilon(\mathbf{x})$ at $\mathbf{x}^{(k)}$ with positive

second order derivative as large as possible. Typically, each component is approximated by a different quadratic polynomial. This determines the adaptive quadratic tangent majorant $\mathcal{Q}^A(\mathbf{x}, \mathbf{x}^{(k)})$ of $\mathcal{J}_\varepsilon(\mathbf{x})$ at $\mathbf{x}^{(k)}$. Thus, $\mathcal{Q}^A(\mathbf{x}, \mathbf{x}^{(k)})$ satisfies

$$\begin{aligned}
\mathcal{Q}^A(\mathbf{x}^{(k)}, \mathbf{x}^{(k)}) &= \mathcal{J}_\varepsilon(\mathbf{x}^{(k)}), && (7)\\
\nabla_{\mathbf{x}} \mathcal{Q}^A(\mathbf{x}^{(k)}, \mathbf{x}^{(k)}) &= \nabla_{\mathbf{x}} \mathcal{J}_\varepsilon(\mathbf{x}^{(k)}), && (8)\\
\mathcal{Q}^A(\mathbf{x}, \mathbf{x}^{(k)}) &\geq \mathcal{J}_\varepsilon(\mathbf{x}) \ \forall \mathbf{x} \in \mathbb{R}^n, && (9)\\
\mathbf{x} &\to \mathcal{Q}^A(\mathbf{x}, \mathbf{x}^{(k)}) \quad \text{is a quadratic functional;} && (10)
\end{aligned}$$

see [20] for details. Here $\nabla_{\mathbf{x}}$ denotes the gradient with respect to $\mathbf{x}$.

The majorant is constructed by evaluating the residual vectors

$$\mathbf{v}^{(k)} = A\mathbf{x}^{(k)} - \mathbf{b}^\delta, \quad \mathbf{u}^{(k)} = L\mathbf{x}^{(k)},$$

which define the *weight vectors*

$$\boldsymbol{\omega}_{\text{fid}}^{A,(k)} = \left(\left(\mathbf{v}^{(k)} \right)^2 + \varepsilon^2 \mathbf{1} \right)^{p/2-1}, \quad \boldsymbol{\omega}_{\text{reg}}^{A,(k)} = \left(\left(\mathbf{u}^{(k)} \right)^2 + \varepsilon^2 \mathbf{1} \right)^{q/2-1},$$

where $\mathbf{1} = [1, 1, \ldots, 1]^T$ and all the operations are meant element-wise. The weight vectors determine the diagonal matrices

$$W_{\text{fid}}^{(k)} = \text{diag}\left(\boldsymbol{\omega}_{\text{fid}}^{A,(k)} \right) \quad \text{and} \quad W_{\text{reg}}^{(k)} = \text{diag}\left(\boldsymbol{\omega}_{\text{reg}}^{A,(k)} \right).$$

The adaptive quadratic tangent majorant of $\mathcal{J}_\varepsilon$ at $\mathbf{x}^{(k)}$ is given by

$$\mathcal{Q}^A(\mathbf{x}, \mathbf{x}^{(k)}) = \frac{1}{2} \left\| \left(W_{\text{fid}}^{(k)} \right)^{1/2} \left(A\mathbf{x} - \mathbf{b}^\delta \right) \right\|_2^2 + \frac{\mu}{2} \left\| \left(W_{\text{reg}}^{(k)} \right)^{1/2} L\mathbf{x} \right\|_2^2 + c,$$

where $c \in \mathbb{R}$ is a constant that is independent of $\mathbf{x}$.

The minimizer $\mathbf{x}^{(k+1)}$ of $\mathcal{Q}^A(\mathbf{x}, \mathbf{x}^{(k)})$ is the next approximate solution of (6). We discuss the computation of an approximation of this minimizer below.

We turn to the construction of the majorant that in [20] is referred to as "fixed." This majorant is constructed by determining quadratic polynomial majorants for each component of $\mathcal{J}_\varepsilon(\mathbf{x})$ at $\mathbf{x}^{(k)}$ with the leading coefficients of all quadratic polynomials chosen to be the same; see [20] for details. This results in simplifications of the computations, when compared to using the adaptive majorant. However, the method so obtained may require more iteration steps to satisfy the stopping criterion.

The *weight vectors* for the fixed majorant are given by

$$\begin{aligned}
\boldsymbol{\omega}_{\text{fid}}^{F,(k)} &= \mathbf{v}^{(k)} \left(\mathbf{1} - \left(\frac{\left(\mathbf{v}^{(k)} \right)^2 + \varepsilon^2 \mathbf{1}}{\varepsilon^2} \right)^{p/2-1} \right), \\
\boldsymbol{\omega}_{\text{reg}}^{F,(k)} &= \mathbf{u}^{(k)} \left(\mathbf{1} - \left(\frac{\left(\mathbf{u}^{(k)} \right)^2 + \varepsilon^2 \mathbf{1}}{\varepsilon^2} \right)^{q/2-1} \right),
\end{aligned} \qquad (11)$$

where all operations are element-wise. We obtain the fixed quadratic tangent majorant

$$\begin{aligned}\mathcal{Q}^F(\mathbf{x},\mathbf{x}^{(k)}) = \tfrac{1}{2}\left(\left\|A\mathbf{x}-\mathbf{b}^\delta\right\|_2^2 - 2\left\langle \boldsymbol{\omega}_{\text{fid}}^{F,(k)}, A\mathbf{x}\right\rangle\right)\\ + \tfrac{\mu}{2}\varepsilon^{q-p}\left(\|L\mathbf{x}\|_2^2 - 2\left\langle \boldsymbol{\omega}_{\text{reg}}^{F,(k)}, L\mathbf{x}\right\rangle\right) + c,\end{aligned}$$

of $\mathcal{J}_\varepsilon$ at $\mathbf{x}^{(k)}$. Here $\langle\cdot,\cdot\rangle$ denotes the standard inner product and the constant $c\in\mathbb{R}$ is independent of $\mathbf{x}$. The functional $\mathcal{Q}^F(\mathbf{x},\mathbf{x}^{(k)})$ satisfies the properties (7)-(10) with $\mathcal{Q}^A(\mathbf{x},\mathbf{x}^{(k)})$ replaced by $\mathcal{Q}^F(\mathbf{x},\mathbf{x}^{(k)})$; see [20] for details.

Minimization step. We describe how to minimize $\mathcal{Q}^A$ and $\mathcal{Q}^F$ when the matrix $A\in\mathbb{R}^{m\times n}$ is large. To reduce the computational effort, we seek to determine approximate solutions in solution subspaces $\mathcal{V}_k$ of dimension $\widehat{k}$, with $k\le\widehat{k}\ll\min\{m,n\}$. Let the columns of the matrix $V_k\in\mathbb{R}^{n\times\widehat{k}}$ form an orthonormal basis for $\mathcal{V}_k$. We determine approximations of the minima of $\mathcal{Q}^A$ and $\mathcal{Q}^F$ of the form

$$\mathbf{x}^{(k+1)} = V_k\mathbf{y}^{(k+1)}, \tag{12}$$

where $\mathbf{y}^{(k+1)}\in\mathbb{R}^{\widehat{k}}$.

Let us first consider the adaptive case. We would like to solve

$$\min_{\mathbf{x}\in\mathcal{V}_k}\frac{1}{2}\left\|\left(W_{\text{fid}}^{(k)}\right)^{1/2}(A\mathbf{x}-\mathbf{b}^\delta)\right\|_2^2 + \frac{\mu}{2}\left\|\left(W_{\text{reg}}^{(k)}\right)^{1/2}L\mathbf{x}\right\|_2^2 \tag{13}$$

and denote the solution by $\mathbf{x}^{(k+1)}$. This is equivalent to computing the solution $\mathbf{y}^{(k+1)}$ of

$$\min_{\mathbf{y}\in\mathbb{R}^{\widehat{k}}}\frac{1}{2}\left\|\left(W_{\text{fid}}^{(k)}\right)^{1/2}(AV_k\mathbf{y}-\mathbf{b}^\delta)\right\|_2^2 + \frac{\mu}{2}\left\|\left(W_{\text{reg}}^{(k)}\right)^{1/2}LV_k\mathbf{y}\right\|_2^2. \tag{14}$$

Introduce the economic QR factorizations

$$\begin{aligned}\left(W_{\text{fid}}^{(k)}\right)^{1/2}AV_k &= Q_AR_A, \quad Q_A\in\mathbb{R}^{m\times\widehat{k}},\ R_A\in\mathbb{R}^{\widehat{k}\times\widehat{k}},\\ \left(W_{\text{reg}}^{(k)}\right)^{1/2}LV_k &= Q_LR_L, \quad Q_L\in\mathbb{R}^{s\times\widehat{k}},\ R_L\in\mathbb{R}^{\widehat{k}\times\widehat{k}},\end{aligned} \tag{15}$$

and compute

$$\mathbf{y}^{(k+1)} = \arg\min_{\mathbf{y}\in\mathbb{R}^{\widehat{k}}}\frac{1}{2}\left\|R_A\mathbf{y}-Q_A^T\left(W_{\text{fid}}^{(k)}\right)^{1/2}\mathbf{b}^\delta\right\|_2^2 + \frac{\mu}{2}\|R_L\mathbf{y}\|_2^2,$$

where the superscript ^{T} denotes transposition. Assuming that

$$\mathcal{N}\left(\left(W_{\text{fid}}^{(k)}\right)^{1/2}AV_k\right)\cap\mathcal{N}\left(\left(W_{\text{fid}}^{(k)}\right)^{1/2}LV_k\right) = \{\mathbf{0}\},$$

which typically holds in applications of interest to us, the solution $\mathbf{y}^{(k+1)}$ is unique. The approximate minimizer of $\mathcal{Q}^A(\mathbf{x},\mathbf{x}^{(k)})$ then is given by (12).

We now enlarge the solution subspace $\mathcal{V}_k$ by including the normalized residual of the normal equations associated with (13). Thus, let

$$\mathbf{r}^{(k+1)} = A^T W_{\text{fid}}^{(k)} \left(A\mathbf{x}^{(k+1)} - \mathbf{b}^\delta \right) + \mu L^T W_{\text{reg}}^{(k)} L\mathbf{x}^{(k+1)}.$$

Then the columns of the matrix

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

furnish an orthonormal basis for the new solution subspace $\mathcal{V}_{k+1}$. We remark that the vector $\mathbf{r}^{(k+1)}$ is proportional to the gradient of $\mathcal{Q}^A(\mathbf{x}, \mathbf{x}^{(k)})$ restricted to $\mathcal{V}_k$ at $\mathbf{x} = \mathbf{x}^{(k+1)}$. We refer to the solution subspace $\mathcal{V}_{k+1} = \text{range}(V_{k+1})$ as a *generalized Krylov subspace*. Note that the computation of $\mathbf{r}^{(k+1)}$ requires only one matrix-vector product with A^T and L^T, since one can use the QR factorizations (15) and the relation (12) to avoid forming matrix-vector products with the matrices A and L. Moreover, we store and update the "skinny" matrices AV_k and LV_k at each iteration to reduce the computational cost. The initial subspace $\mathcal{V}_1$ is usually chosen to contain a few selected vectors and to be of small dimension. A common choice is $\mathcal{V}_1 = \mathsf{span}\{A^T\mathbf{b}^\delta\}$. Then $\widehat{k} = k$. We will use this choice in the computed examples reported in Section 4.

Summarizing, each iteration of the adaptive approach requires one matrix-vector product evaluation with each one of the matrices A, L, A^T, and L^T, as well as the computation of economic QR factorizations of two tall and skinny matrices, whose column numbers increase by one with each iteration. The latter computations can be quite demanding if the matrices A and L are large and many iterations are required. The algorithm requires storage of the three matrices V_k, AV_k, and LV_k. In addition, storage of some representations of the matrices A and L is needed.

We turn to the fixed approach. The weight vectors are now given by (11), and we would like to solve the minimization problem

$$\min_{\mathbf{x}\in\mathcal{V}_k} \frac{1}{2} \left(\left\| A\mathbf{x} - \mathbf{b}^\delta \right\|_2^2 - 2 \left\langle \boldsymbol{\omega}_{\text{fid}}^{F,(k)}, A\mathbf{x} \right\rangle \right) + \frac{\eta}{2} \left(\|L\mathbf{x}\|_2^2 - 2 \left\langle \boldsymbol{\omega}_{\text{reg}}^{F,(k)}, L\mathbf{x} \right\rangle \right) \quad (16)$$

for $\mathbf{x}^{(k+1)}$, where $\eta = \mu\varepsilon^{q-p}$. This problem can be expressed as

$$\min_{\mathbf{y}\in\mathbb{R}^{\widehat{k}}} \left\| AV_k\mathbf{y} - \mathbf{b}^\delta - \boldsymbol{\omega}_{\text{fid}}^{F,(k)} \right\|_2^2 + \eta \left\| LV_k\mathbf{y} - \boldsymbol{\omega}_{\text{reg}}^{F,(k)} \right\|_2^2. \quad (17)$$

The solution $\mathbf{y}^{(k+1)}$ of (17) yields the solution $\mathbf{x}^{(k+1)} = V_k\mathbf{y}^{(k+1)}$ of (16).

Introduce the economic QR factorizations

$$\begin{aligned} AV_k &= Q_A R_A, \quad Q_A \in \mathbb{R}^{m\times\widehat{k}},\ R_A \in \mathbb{R}^{\widehat{k}\times\widehat{k}}, \\ LV_k &= Q_L R_L, \quad Q_L \in \mathbb{R}^{s\times\widehat{k}},\ R_L \in \mathbb{R}^{\widehat{k}\times\widehat{k}}. \end{aligned}$$

Substituting these factorizations into (17) yields

$$\mathbf{y}^{(k+1)} = \arg\min_{\mathbf{y}\in\mathbb{R}^{\widehat{k}}} \left\| \begin{bmatrix} R_A \\ \sqrt{\eta} R_L \end{bmatrix} \mathbf{y} - \begin{bmatrix} Q_A^T \left(\mathbf{b}^\delta + \boldsymbol{\omega}_{\text{fid}}^{F,(k)} \right) \\ \sqrt{\eta} Q_L^T \boldsymbol{\omega}_{\text{reg}}^{F,(k)} \end{bmatrix} \right\|_2^2.$$

Once we have computed $\mathbf{y}^{(k+1)}$ and $\mathbf{x}^{(k+1)}$, we enlarge the solution subspace by including the residual

$$\mathbf{r}^{(k+1)} = A^T \left(A\mathbf{x}^{(k+1)} - \left(\mathbf{b}^\delta + \boldsymbol{\omega}_{\text{fid}}^{F,(k)} \right) \right) + \eta L^T \left(L\mathbf{x}^{(k+1)} - \boldsymbol{\omega}_{\text{reg}}^{F,(k)} \right)$$

of the normal equations associated with (16). Thus, let $\mathbf{v}_{\text{new}} = \mathbf{r}^{(k+1)} / \left\| \mathbf{r}^{(k+1)} \right\|_2$. Then the columns of the matrix $V_{k+1} = [V_k, \mathbf{v}_{\text{new}}]$ form an orthonormal basis for the solution subspace $\mathcal{V}_{k+1}$. We remark that the residual is proportional to the gradient of $\mathcal{Q}^F(\mathbf{x}, \mathbf{x}^{(k)})$ restricted to $\mathcal{V}_k$ at $\mathbf{x} = \mathbf{x}^{(k+1)}$.

Note that, differently from (14), the least-squares problem (17) does not have a diagonal scaling matrix. We therefore may compute the QR factorizations of AV_{k+1} and LV_{k+1} by updating the QR factorizations of AV_k and LV_k, respectively. This reduces the computational work and leads to that each new iteration with the fixed approach is cheaper to carry out than with the adaptive approach. Updating formulas for the QR factorization can be found in [11, 20].

Each iteration with the fixed approach requires one matrix-vector product evaluation with each one of the matrices A, L, A^T, and L^T, similarly as for the adaptive approach. Moreover the memory requirements of the fixed and adaptive approaches are essentially the same.

The memory requirement for both the adaptive and fixed approaches outlined grows linearly with the number of iterations. It follows that when the matrix A is large, the memory requirement may be substantial when many iterations are required to satisfy the stopping criterion. This could be a difficulty on computers with fairly little fast memory. Moreover, the arithmetic cost of computing QR factorizations in the adaptive approach and for updating QR factorizations in the fixed approach grows quadratically and linearly, respectively, with the number of iterations. Therefore, curtailing the growth of the dimension of the solution subspace used reduces both the memory requirement and the arithmetic work per iteration. The following section discusses an approach to achieve this.

3 Restarted generalized Krylov subspace methods

Assume that only a very limited amount of fast computer memory is available. Then it may be expedient to bound the dimension of the solution subspaces used by the MM methods by restarting the algorithms. Let $K_{\max}$ denote the maximal permitted dimension of the solution subspaces and assume that for a certain k, it holds that $V_k \in \mathbb{R}^{n \times K_{\max}}$. To avoid that the dimension of the solution subspaces increase further, we restart the MM algorithms. In detail, let $\widetilde{\mathbf{x}} = \mathbf{x}^{(k)}$. We would like to determine $\mathbf{x}^{(k+1)}$ as

$$\mathbf{x}^{(k+1)} = \widetilde{V}_{k+1} \mathbf{y}^{(k+1)},$$

where $\widetilde{V}_{k+1} \in \mathbb{R}^{n \times \widetilde{k}}$ has orthonormal columns with $\widetilde{k} < K_{\max}$ and $\widetilde{\mathbf{x}} \in \text{range}(\widetilde{V}_{k+1})$. In the computed examples reported in Section 4, we let $\widetilde{k} = 1$,

i.e., $\widetilde{V}_{k+1} = \widetilde{\mathbf{x}}/\left\|\widetilde{\mathbf{x}}\right\|_2$. To achieve this, we modify the adaptive and fixed MM algorithms as described by Algorithms 1 and 2.

Algorithm 1: A-MM-GKS-Restarted

Let $A \in \mathbb{R}^{m\times n}$, $\mathbf{b}^\delta \in \mathbb{R}^m$. Let $\mu > 0$ be a fixed parameter and $L \in \mathbb{R}^{s\times n}$ be such that $\mathcal{N}(A) \cap \mathcal{N}(L) = \{\mathbf{0}\}$. Fix $0 < p, q \leq 2$, a maximum number of iterations K, a maximum dimension $K_{\max}$, a smoothing parameter $\varepsilon > 0$, and a tolerance $\tau > 0$. Let $\mathbf{x}^{(0)}$ be an initial guess for $\mathbf{x}^\dagger$;
$V_0 = A^T\mathbf{b}^\delta/\left\|A^T\mathbf{b}^\delta\right\|_2$;
Compute and store AV_0 and LV_0;
$\mathbf{v}^{(0)} = A\mathbf{x}^{(0)} - \mathbf{b}^\delta$;
$\mathbf{u}^{(0)} = L\mathbf{x}^{(0)}$;
for $k = 0, 1, \ldots, K$ **do**
  $\boldsymbol{\omega}_{\text{fid}}^{A,(k)} = \left(\left(\mathbf{v}^{(k)}\right)^2 + \varepsilon^2\right)^{p/2-1}$;
  $\boldsymbol{\omega}_{\text{reg}}^{A,(k)} = \left(\left(\mathbf{u}^{(k)}\right)^2 + \varepsilon^2\right)^{q/2-1}$;
  $W_{\text{fid}}^{(k)} = \text{diag}\left(\boldsymbol{\omega}_{\text{fid}}^{A,(k)}\right)$;
  $W_{\text{reg}}^{(k)} = \text{diag}\left(\boldsymbol{\omega}_{\text{reg}}^{A,(k)}\right)$;
  Compute the QR factorizations $\begin{cases} \left(W_{\text{fid}}^{(k)}\right)^{1/2} AV_k = Q_A R_A \\ \left(W_{\text{reg}}^{(k)}\right)^{1/2} LV_k = Q_L R_L \end{cases}$;
  $\mathbf{y}^{(k+1)} = \arg\min_{\mathbf{y}} \frac{1}{2}\left\|R_A\mathbf{y} - Q_A^T\left(W_{\text{fid}}^{(k)}\right)^{1/2}\mathbf{b}^\delta\right\|_2^2 + \frac{\mu}{2}\left\|R_L\mathbf{y}\right\|_2^2$;
  if $k > 1$ *&* $\left\|\mathbf{y}^{(k+1)} - \mathbf{y}^{(k)}\right\|_2 < \tau\left\|\mathbf{y}^{(k)}\right\|_2$ **then**
    break;
  end
  if $k + 1 \equiv 0 \ mod \ K_{\max}$ **then**
    $\mathbf{v}^{(k+1)} = AV_k\mathbf{y}^{(k+1)} - \mathbf{b}^\delta$;
    $\mathbf{u}^{(k+1)} = LV_k\mathbf{y}^{(k+1)}$;
    $\widetilde{\mathbf{x}} = V_k\mathbf{y}^{(k+1)}$;
    Set $V_{k+1} = \widetilde{\mathbf{x}}/\left\|\widetilde{\mathbf{x}}\right\|_2$;
    Compute and store AV_{k+1};
    Compute and store LV_{k+1};
  else
    $\mathbf{r}^{(k+1)} = A^T W_{\text{fid}}^{(k)}\left(AV_k\mathbf{y}^{(k+1)} - \mathbf{b}^\delta\right) + \mu L^T W_{\text{reg}}^{(k)}\left(LV_k\mathbf{y}^{(k+1)}\right)$;
    $\mathbf{v}_{\text{new}} = \mathbf{r}^{(k+1)}/\left\|\mathbf{r}^{(k+1)}\right\|_2$;
    $V_{k+1} = [V_k, \mathbf{v}_{\text{new}}]$;
    $AV_{k+1} = [AV_k, A\mathbf{v}_{\text{new}}]$;
    $LV_{k+1} = [LV_k, L\mathbf{v}_{\text{new}}]$;
    $\mathbf{v}^{(k+1)} = AV_k\mathbf{y}^{(k+1)} - \mathbf{b}^\delta$;
    $\mathbf{u}^{(k+1)} = LV_k\mathbf{y}^{(k+1)}$;
  end
end

We turn to convergence properties of Algorithms 1 and 2. Since the analysis of the two algorithms is identical and does not depend on the construction of the quadratic tangent majorant, we simply denote this functional by $\mathcal{Q}(\mathbf{x}, \mathbf{x}^k)$.

Algorithm 2: F-MM-GKS-Restarted

Let $A \in \mathbb{R}^{m\times n}$, $\mathbf{b}^\delta \in \mathbb{R}^m$. Let $\mu > 0$ be a fixed parameter and $L \in \mathbb{R}^{s\times n}$ be such that $\mathcal{N}(A) \cap \mathcal{N}(L) = \{\mathbf{0}\}$. Fix $0 < p, q \leq 2$, a maximum number of iterations K, a maximum dimension $K_{\max}$, a smoothing parameter $\varepsilon > 0$, and a tolerance $\tau > 0$. Let $\mathbf{x}^{(0)}$ be an initial guess for $\mathbf{x}^\dagger$;
$V_0 = A^T\mathbf{b}^\delta / \left\|A^T\mathbf{b}^\delta\right\|_2$;
Compute and store AV_0 and LV_0;
Compute the QR factorizations $\begin{cases} AV_0 = Q_A R_A \\ LV_0 = Q_L R_L \end{cases}$;
$\mathbf{v}^{(0)} = A\mathbf{x}^{(0)} - \mathbf{b}^\delta$;
$\mathbf{u}^{(0)} = L\mathbf{x}^{(0)}$;
$\eta = \frac{\mu\varepsilon^{q-2}}{\varepsilon^{p-2}}$;
for $k = 0, 1, \ldots, K$ **do**
  $\boldsymbol{\omega}_{\text{fid}}^{F,(k)} = \mathbf{v}^{(k)}\left(1 - \left(\frac{(\mathbf{v}^{(k)})^2+\varepsilon^2}{\varepsilon^2}\right)^{p/2-1}\right)$;
  $\boldsymbol{\omega}_{\text{reg}}^{F,(k)} = \mathbf{u}^{(k)}\left(1 - \left(\frac{(\mathbf{u}^{(k)})^2+\varepsilon^2}{\varepsilon^2}\right)^{q/2-1}\right)$;
  $\mathbf{y}^{(k+1)} = \arg\min_{\mathbf{y}} \left\|AV_k\mathbf{y} - \mathbf{b}^\delta - \boldsymbol{\omega}_{\text{fid}}^{F,(k)}\right\|_2^2 + \eta\left\|LV_k\mathbf{y} - \boldsymbol{\omega}_{\text{reg}}^{F,(k)}\right\|_2^2$;
  if $k > 1$ *&* $\left\|\mathbf{y}^{(k+1)} - \mathbf{y}^{(k)}\right\|_2 < \tau\left\|\mathbf{y}^{(k)}\right\|_2$ **then**
    break;
  end
  if $k + 1 \equiv 0 \ mod\ K_{\max}$ **then**
    $\widetilde{\mathbf{x}} = V_k\mathbf{y}^{(k+1)}$;
    $\mathbf{v}^{(k+1)} = AV_k\mathbf{y}^{(k+1)} - \mathbf{b}^\delta$;
    $\mathbf{u}^{(k+1)} = LV_k\mathbf{y}^{(k+1)}$;
    $V_{k+1} = \widetilde{\mathbf{x}} / \left\|\widetilde{\mathbf{x}}\right\|_2$;
    Compute and store AV_{k+1};
    Compute and store LV_{k+1};
    Compute the QR factorizations $\begin{cases} AV_{k+1} = Q_A R_A \\ LV_{k+1} = Q_L R_L \end{cases}$;
  else
    $\mathbf{r}^{(k+1)} = A^T\left(AV_k\mathbf{y}^{(k+1)} - \mathbf{b}^\delta - \boldsymbol{\omega}_{\text{fid}}^{F,(k)}\right) + \eta L^T\left(LV_k\mathbf{y}^{(k+1)} - \boldsymbol{\omega}_{\text{reg}}^{F,(k)}\right)$;
    $\mathbf{v}_{\text{new}} = \mathbf{r}^{(k+1)} / \left\|\mathbf{r}^{(k+1)}\right\|_2$;
    $V_{k+1} = [V_k, \mathbf{v}_{\text{new}}]$;
    $AV_{k+1} = [AV_k, A\mathbf{v}_{\text{new}}]$;
    $LV_{k+1} = [LV_k, L\mathbf{v}_{\text{new}}]$;
    $\mathbf{v}^{(k+1)} = AV_k\mathbf{y}^{(k+1)} - \mathbf{b}^\delta$;
    $\mathbf{u}^{(k+1)} = LV_k\mathbf{y}^{(k+1)}$;
    Update Q_A, Q_L, R_A, and R_L as described in [11, 20];
  end
end

We introduce some notation that will be used in the following. Let

$$\mathcal{V}_k = \mathsf{span}\left\{\mathbf{v}_1, \ldots, \mathbf{v}_{\hat{k}}\right\}, \qquad V_k = [\mathbf{v}_1, \ldots, \mathbf{v}_{\hat{k}}],$$

and define the indicator function ι_k of $\mathcal{V}_k$, i.e.,

$$\iota_k(\mathbf{x}) = \begin{cases} 0, & \mathbf{x} \in \mathcal{V}_k, \\ +\infty, & \text{otherwise.} \end{cases}$$

We can write

$$\mathbf{x}^{(k+1)} = \arg\min_{\mathbf{x}\in\mathbb{R}^n} \left(\mathcal{Q}(\mathbf{x}, \mathbf{x}^{(k)}) + \iota_k(\mathbf{x}) \right). \tag{18}$$

Note that, by construction, $\mathbf{x}^{(k)} \in \mathcal{V}_{k-1} \cap \mathcal{V}_k$ for all $k = 1, 2, \dots$, since at each iteration we either expand the solution subspace, or restart it including the previous iterate in it.

We first show that the sequence $\left\{ \mathcal{J}_\varepsilon \left(\mathbf{x}^{(k)} \right) \right\}$ is monotonically decreasing.

Proposition 1 *Let $\mathbf{x}^{(k)}$, $k = 1, 2, \dots$, denote the iterates generated by either Algorithm 1 or Algorithm 2. Assume that condition* (3) *holds. Then*

$$\mathcal{J}_\varepsilon \left(\mathbf{x}^{(k+1)} \right) \leq \mathcal{J}_\varepsilon \left(\mathbf{x}^{(k)} \right).$$

Proof Recall that $\mathbf{x}^{(k)} \in \mathcal{V}_{k-1} \cap \mathcal{V}_k$ for all $k = 1, 2, \dots$ and, therefore, $\iota_k \left(\mathbf{x}^{(k)} \right) = \iota_k \left(\mathbf{x}^{(k+1)} \right) = 0$. We obtain

$$\begin{aligned}
\mathcal{J}_\varepsilon \left(\mathbf{x}^{(k+1)} \right) &= \mathcal{J}_\varepsilon \left(\mathbf{x}^{(k+1)} \right) + \iota_k \left(\mathbf{x}^{(k+1)} \right) \\
&\overset{(a)}{\leq} \mathcal{Q} \left(\mathbf{x}^{(k+1)}, \mathbf{x}^{(k)} \right) + \iota_k \left(\mathbf{x}^{(k+1)} \right) \\
&\overset{(b)}{\leq} \mathcal{Q} \left(\mathbf{x}^{(k)}, \mathbf{x}^{(k)} \right) + \iota_k \left(\mathbf{x}^{(k)} \right) \\
&= \mathcal{Q} \left(\mathbf{x}^{(k)}, \mathbf{x}^{(k)} \right) = \mathcal{J}_\varepsilon \left(\mathbf{x}^{(k)} \right),
\end{aligned}$$

where (a) follows from the fact that $\mathcal{Q}(\mathbf{x}, \mathbf{x}^{(k)})$ is a majorant of $\mathcal{J}_\varepsilon(\mathbf{x})$ and (b) follows from (18).

Remark 1 It is immediate to see that Proposition 1 carries over to the non-restarted methods, A-MM-GKS and F-MM-GKS, described in [20]. It is shown in [20] that

$$\mathcal{J}_\varepsilon \left(\mathbf{x}^{(k+1)} \right) \leq \mathcal{J}_\varepsilon \left(\mathbf{x}^{(k)} \right)$$

for k large enough, i.e., for k such that $\mathcal{V}_k = \mathbb{R}^n$, but no results are provided for small values of k. Proposition 1 shows that $\mathcal{J}_\varepsilon \left(\mathbf{x}^{(k)} \right)$ decreases monotonically for all k in the A-MM-GKS and F-MM-GKS methods and, thus, complements the analysis presented in [20].

We are in the position to show our main result.

Theorem 1 *Let $\mathbf{x}^{(k)}$, $k = 1, 2, \dots$, denote the iterates generated by either Algorithm 1 or Algorithm 2. Assume that condition* (3) *holds. Then there exists a convergent subsequence $\{\mathbf{x}^{(k_j)}\}$.*

Proof By Proposition 1 we have that, for all k,

$$\mathcal{J}_\varepsilon \left(\mathbf{x}^{(k)} \right) \leq \mathcal{J}_\varepsilon \left(\mathbf{x}^{(0)} \right) =: C.$$

Since $\Phi_{s,\varepsilon}(t) \leq |t|^s$, it follows that

$$\left\| A\mathbf{x}^{(k)} - \mathbf{b}^\delta \right\|_p^p \leq \sum_{j=1}^{m} \Phi_{p,\varepsilon}\left(\left(A\mathbf{x}^{(k)} - \mathbf{b}^\delta \right)_j \right) \leq p\mathcal{J}_\varepsilon\left(\mathbf{x}^{(k)} \right) \leq pC$$

and

$$\left\| L\mathbf{x}^{(k)} \right\|_q^q \leq \sum_{j=1}^{s} \Phi_{q,\varepsilon}\left(\left(L\mathbf{x}^{(k)} \right)_j \right) \leq \frac{q}{\mu}\mathcal{J}_\varepsilon\left(\mathbf{x}^{(k)} \right) \leq \frac{qC}{\mu}.$$

It now follows from [6, Lemma 1] that there is a constant $\widetilde{C} \in \mathbb{R}$ such that, for all k,

$$\left\| A\mathbf{x}^{(k)} - \mathbf{b}^\delta \right\|_2^2 + \left\| L\mathbf{x}^{(k)} \right\|_2^2 \leq \widetilde{C}.$$

Since we assumed condition (3) to hold, there is a constant $\widehat{C}$ such that

$$\left\| \mathbf{x}^{(k)} \right\|_2^2 \leq \widehat{C} \quad \text{for all } k = 1, 2, \ldots \ .$$

Because the sequence $\{\mathbf{x}^{(k)}\}$ is uniformly bounded, it has a convergent subsequence $\left\{\mathbf{x}^{(k_j)}\right\}$.

In extensive numerical experiments, we have never observed the need to choose a convergence subsequence. Our experience suggests that the iterates $\mathbf{x}^{(k)}$, $k = 1, 2, \ldots$, generally are convergent.

4 Numerical examples

This section presents a few numerical examples that illustrate the performance of the limited memory algorithms described and compares these algorithms to their standard counterparts presented in [20]. The computed examples show the algorithms of this paper to determine approximate solutions of similar quality as the standard algorithms in [20], but to require significantly less computer memory and less execution time. The regularization matrix L is chosen to be a discretization of the gradient. This is justified by the fact that many natural images have a sparse gradient; [24] for discussions of this regularization approach.

It is essential that the regularization parameter μ be chosen properly for the accurate approximation of the desired approximate solution $\mathbf{x}^*$ of (1). A comparison of several techniques for determining μ has recently been presented in [7]. In this paper, we consider two strategies for determining μ: for Gaussian noise we apply the Discrepancy Principle (DP) [8]; for other kinds of noise we use Generalized Cross Validation (GCV) [9]. The DP requires that a fairly accurate estimate of the norm of the noise be available, while application of GCV does not require this information. However, the DP allows for a theoretical analysis of the regularization properties of the method, while GCV is a so-called heuristic method that may fail for certain problems; see [22,23,26] for

discussions on heuristic methods. Finally, we would like to mention that the parameter μ also can be determined by Cross Validation (CV) and Modified Cross Validation (MCV) as described in [8]. We will not illustrate the use of the latter approaches here, since the results easily can be inferred from the ones presented. In fact, the CV and MCV methods apply the MM methods several times with different subvectors of $\mathbf{b}^\delta$ and submatrices of A, and compare the computed results to determine a suitable value of the regularization parameter μ. Therefore, the memory usage of the CV and MCV methods is the same as when applying MM methods for a fixed parameter μ.

For our numerical experiments, we consider space-invariant image deblurring problems that are modeled by a Fredholm integral equation of the first kind. Thus, the problems solved are deconvolution problems in two space-dimensions. The kernel of the integral often is referred to as a Point Spread Function (PSF); see, e.g., [19] for details on image deblurring. The matrix A in (1) is a discretization of the integral equation and typically is very ill-conditioned; it may be numerically rank-deficient.

In the examples we assume that an exact gray scale image is available, whose pixel values are represented by the vector $\mathbf{x}_{\text{exact}}$. The entries of this vector are integers in the interval $[0, 255]$. A noise vector $\boldsymbol{\eta}$ with specified properties, and the vector $\mathbf{b}^\delta$ which represents a blur- and noise-contaminated image, is generated by (1) with $\mathbf{x}$ replaced by $\mathbf{x}_{\text{exact}}$.

We measure the quality of the computed approximations $\mathbf{x}^*$ of the image $\mathbf{x}_{\text{exact}}$ by the Peak Signal to Noise Ratio (PSNR) defined by

$$\text{PSNR}(\mathbf{x}^*) = 20 \log_{10} \left(\frac{255\sqrt{n}}{\|\mathbf{x}^* - \mathbf{x}_{\text{exact}}\|_2} \right).$$

Here we assume that the entries of the vectors that represent images are in the interval $[0, 255]$.

We set the maximum number of iteration to 200, the tolerance τ in Algorithms 1 and 2 to $\tau = 10^{-4}$, and the smoothing parameter ε in (5) to 1; see [8] for comments on the choice of ε for image restoration problems. These parameter values also are used for the "standard" implementations described in [20]. For all methods, we use the initializations $\mathbf{x}^{(0)} = A^T \mathbf{b}^\delta$ and $V_0 = \mathbf{x}^{(0)} / \|\mathbf{x}^{(0)}\|_2$. Finally, we set $K_{\max} = 30$ in the restarted methods.

All the numerical experiments reported were carried out in MATLAB 2021b running on a laptop computer with an AMD Ryzen 7 5800HS CPU and 16GB of RAM.

Satellite. This example considers the restoration of a blurred and noise-contaminated version of the Satellite image in Fig. 1(a). We blur this image with out-of-focus blur; the associated PSF is depicted in Fig. 1(b). Then we add white Gaussian noise such that $\|\boldsymbol{\eta}\|_2 = 0.02 \|\mathbf{b}\|_2$, i.e., we add 2% white Gaussian noise, to obtain the data vector $\mathbf{b}^\delta$ shown in Fig. 1(c). The boundaries are cropped of the image to simulate real data. Since the image is astronomical, we impose zero Dirichlet boundary conditions; see, e.g., [12,19] for discussions on boundary conditions.

We apply the DP to determine the regularization parameter. Following [8], we use the DP with the fixed approach. To ensure that it is possible to satisfy the DP in the first iterations of the MM method we define V_0 to be the basis of the Krylov subspace

$$\mathcal{K}_{20}(A^TA, A^T\mathbf{b}^\delta) = \mathsf{span}\left\{A^T\mathbf{b}^\delta, (A^TA)A^T\mathbf{b}^\delta, \ldots, , (A^TA)^{19}A^T\mathbf{b}^\delta\right\}$$

obtained with the Golub-Kahan algorithm; see, e.g., [17] for more details. Therefore, at iteration k the dimension of the GKS will be $k + 20$.

Note that we can apply GCV only if, at each iteration, the ℓ^2-ℓ^2 problem to be solved is of the form

$$\arg\min_{\mathbf{y}} \left\{ \frac{1}{2} \left\| \widehat{A}\mathbf{y} - \widehat{\mathbf{b}} \right\|_2^2 + \frac{\gamma}{2} \left\| \widehat{L}\mathbf{y} \right\|_2^2 \right\}$$

for some matrices $\widehat{A}$ and $\widehat{L}$. Therefore, we cannot apply GCV with the fixed approach; see [9].

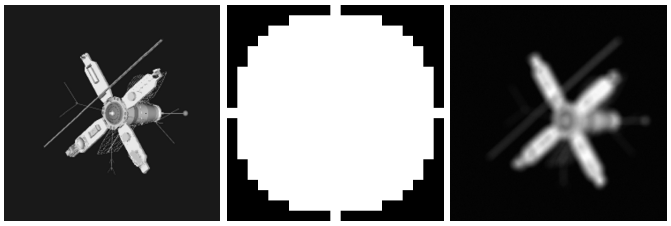
Computed results are reported in Table 2. We observe that the limited memory approach provides very similar results as the standard approach from [20] in terms of the quality of the computed restoration both when the regularization parameter is hand-tuned or chosen with GCV or the DP. However, the restarted methods require significantly less computation time and computer memory.

When the regularization parameter is fixed, the CPU times for the fixed MM method when using the limited memory algorithm and the standard one are almost the same, even though the dimension of the solution subspace is much smaller for the limited memory algorithm. This is due to the fact that at each iteration, the QR factorizations of AV_k and LV_k are updated and do not have to be recomputed from scratch. Therefore, the cost of the two algorithms is about the same. However, when the regularization parameter is determined by the DP, we can observe that the computational cost of the standard approach from [20] is much higher than for Algorithm 2. We compute the regularization parameter for the latter approach by first updating the economical QR factorizations of AV_k and LV_k, and then calculate the GSVD of the pair of upper triangular matrices so obtained. This is a fairly inexpensive way to evaluate the GSVD of the matrix pair $\{AV_{k+1}, LV_{k+1}\}$. Nevertheless, Table 2 shows the limited memory approach to be faster because of the reduced dimension of the solution subspace used. We remark that the GCV method cannot be used to determine the regularization parameter in the presence of impulse noise without preprocessing; see [9] for details. Fig. 2 displays computed restorations.

Finally, Fig. 3 displays the number of bytes used by MATLAB, versus the iteration number, for the standard and restarted approaches to the adaptive and fixed MM methods. We observe that the restarted approaches reduce the memory requirement significantly for both the adaptive and fixed MM methods. As expected, the curves for the restarted methods oscillate and are bounded from above.

Table 1: Role of τ in the Satellite example. We report the PSNR, stopping iteration, and CPU time for A-MM-GKS with GCV and F-MM-GKS with DP for different values of τ in the stopping criterion.

τ	A-MM-GKS (GCV)			F-MM-GKS (DP)		
	PSNR	Iter.	CPU time	PSNR	Iter.	CPU time
10^{-4}	29.661	200	491.81	31.819	200	83.755
$5 \cdot 10^{-4}$	29.661	200	496.23	31.513	101	27.025
10^{-3}	30.211	120	149.298	30.927	52	11.5401
$5 \cdot 10^{-3}$	28.711	24	6.130	28.345	2	1.6194
10^{-2}	22.359	2	0.2754	28.345	2	1.5937



(a) (b) (c)

Fig. 1: Satellite test case: (a) True image (490×490 pixels), (b) PSF (27×27 pixels), (c) Blurred and noisy image with 2% of white Gaussian noise (490×490 pixels).

We now briefly discuss the role of τ in the stopping criterion, and consider the A-MM-GKS with GCV and F-MM-GKS with DP for this analysis. The results for the other methods are similar and we therefore do not report them here. We consider five values of τ, namely $\tau \in \{10^{-4},\ 5 \cdot 10^{-4},\ 10^{-3},\ 5 \cdot 10^{-3},\ 10^{-2}\}$. For each value of τ, we run the two algorithms and report in Table 1 the obtained PSNR, stopping iteration, and CPU time. We can observe that, as expected, as τ increases the number of iterations required to satisfy the stopping criterion decreases. However, we note that, in most cases, the PSNR decreases significantly. Vast computational experience suggests that this is usually the case for the MM method and that a value of τ larger than 10^{-4} usually stops the iterations too soon, and results in poor reconstructions of the desired solution.

Cameraman. We consider the restoration of a blur- and noise-contaminated version of the exact image shown in panel (a) of Fig. 4. We blur this image using the PSF shown in panel (b); the blur simulates hand-shaking. Then we add 25% of impulse noise (see below) to obtain the blurred and noisy image depicted in panel (c). The boundaries are cropped to simulate realistic data. Since the image is generic, we impose reflexive boundary conditions. Due to the fact that the image that we would like to restore is contaminated by impulse noise, we cannot use the DP.

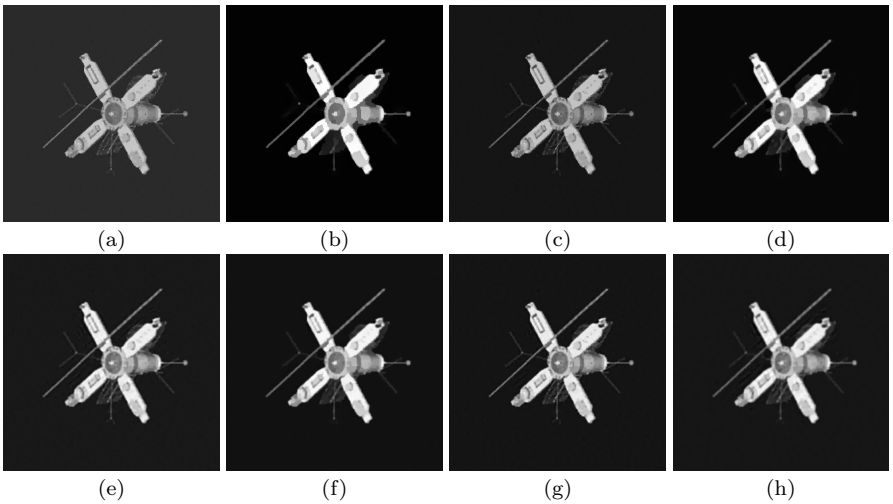


Fig. 2: Satellite test case reconstructions: (a) A-MM-GKS (hand-tuned parameter), (b) A-MM-GKS (GCV parameter), (c) A-MM-GKS Restarted (hand-tuned parameter), (d) A-MM-GKS Restarted (GCV parameter), (e) F-MM-GKS (hand-tuned parameter), (f) F-MM-GKS (DP parameter), (g) F-MM-GKS Restarted (hand-tuned parameter), (h) F-MM-GKS Restarted (DP parameter).

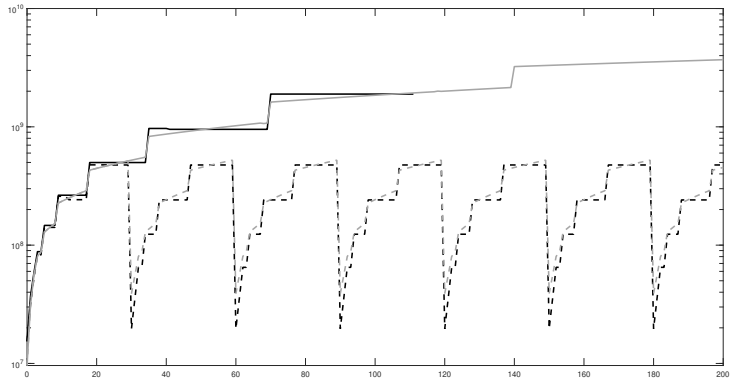


Fig. 3: Satellite test case memory usage plots. The horizontal axis reports the iteration number, and the vertical axis the number of bytes used by the algorithms (hand-tuned parameters). The solid lines depicts the results of the standard method, the dashed line the limited memory approach, the gray lines are refereed to the adaptive majorant, while the black ones are for the fixed majorant.

Table 2 reports results for this example. When the parameter μ is hand-tuned, the quality of the restorations determined by the MM methods and the restarted methods are high and close. The adaptive limited memory MM method combined with GCV is fast, but determines a slightly worse restoration than its "standard" counterpart. Nevertheless, the computed approximation is still very accurate. For the adaptive approach the restarted algorithms are significantly faster than the A-MM-GKS method. However, for the fixed approach, as we discussed above, the computational times are comparable. In any case, the restarted F-MM-GKS method is still computationally interesting as it requires less memory. Figure 5 displays the computed restorations; the PSNR-values reported in Table 2 are in agreement with visual perception.

To motivate our choices of p and q, we plot in Fig. 6 the relative restoration error

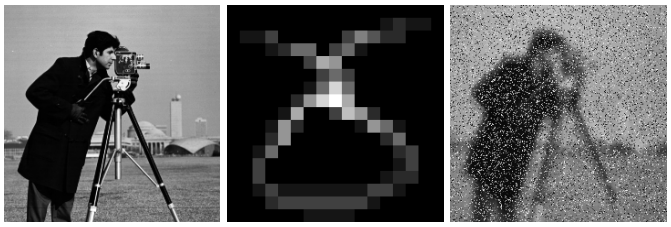
$$\mathrm{RRE}(\mathbf{x}^*) = \frac{\|\mathbf{x}^* - \mathbf{x}_{\mathrm{exact}}\|_2}{\|\mathbf{x}_{\mathrm{exact}}\|_2}$$

obtained for several choices of p and q with the A-MM-GKS restarted method, where μ was selected with the GCV. We can observe that a clear minimum of the RRE can be found when both p and q are smaller than 1. Moreover, we note that in general the RRE decreases with q for fixed $p < 1$. These observations are consistent with results presented in [3, 8, 20].

We now would like to discuss how the choices of p and q influence the computational cost of the algorithm. To do so we fix $\mu = 1$ and run the A-MM-GKS-Restarted and F-MM-GKS-Restarted methods for several values of p and q, and record the number of iterations required to reach numerical convergence. We recall that we stop the iterations as soon as

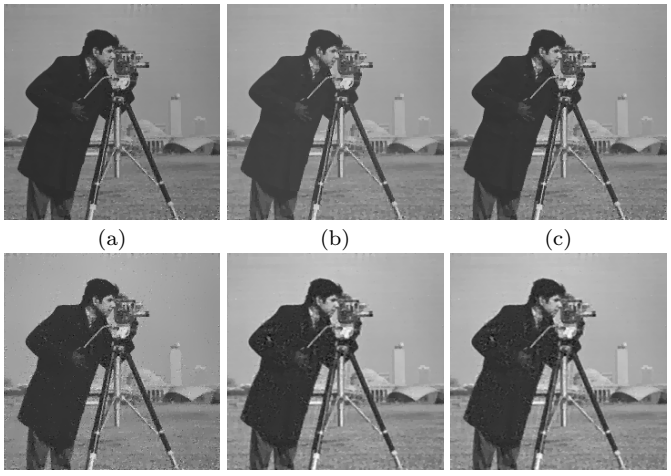
$$\frac{\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}\right\|_2}{\left\|\mathbf{x}^{(k)}\right\|_2} \leq \tau,$$

where we set $\tau = 10^{-4}$ in our experiments. We denote the stopping index by k_{stop}. Fig. 7 displays the computed results. We can see that both methods behave similarly, although, as it was pointed out in [20], the A-MM method usually requires fewer iterations to converge. The figure shows both methods to require the fewest number of iterations to reach numerical convergence when $p = q = 2$. When either p or q are decreased, the number of iterations required generally increases. Moreover, for the fixed approach, we can observe that when both p and q are small enough, k_{stop} is smaller than when one of the parameters p or q is close to 2 and the other one is close to 0. This may be due to the fact that the fixed approach for small values of p or q converges very slowly and, therefore, the stopping criterion may give a k_{stop}-value that is too small. This suggests that the value of τ can be reduced when p or q are close to 0. We would like to point out, however, that in our experience, even if the iterations are terminated too early, the quality of the computed reconstructions usually is satisfactory. The same behavior can be observed, to a smaller extent, in the adaptive case.



(a) (b) (c)

Fig. 4: Cameraman test case: (a) True image (238×238 pixels), (b) PSF (17×17 pixels), (c) Blurred and noisy image with 25% of impulse noise (238 × 238 pixels).



(a) (b) (c)

(d) (e) (f)

Fig. 5: Cameraman test case reconstructions: (a) A-MM-GKS (hand-tuned parameter), (b) A-MM-GKS (GCV parameter), (c) A-MM-GKS Restarted (hand-tuned parameter), (d) A-MM-GKS Restarted (GCV parameter), (e) F-MM-GKS (hand-tuned parameter), (f) F-MM-GKS Restarted (hand-tuned parameter).

Peppers. Our last example is concerned with the restoration of a contaminated version of the peppers image in Fig. 8(a). We blur this image by motion blur defined by the PSF depicted in Fig. 8(b), add a mixture of 1% white Gaussian noise and 10% impulse noise, and crop the boundaries. This yields the blur- and noise-contaminated image shown in Fig. 8(c), which we would like to restore. Since the image is generic, we assume reflexive boundary conditions. Similarly as above, we cannot apply the DP to determine the regularization parameter, because the noise contains impulse noise. We therefore use GCV to determine a suitable value of this parameter.

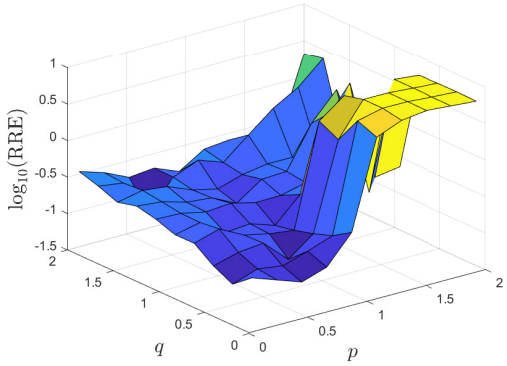
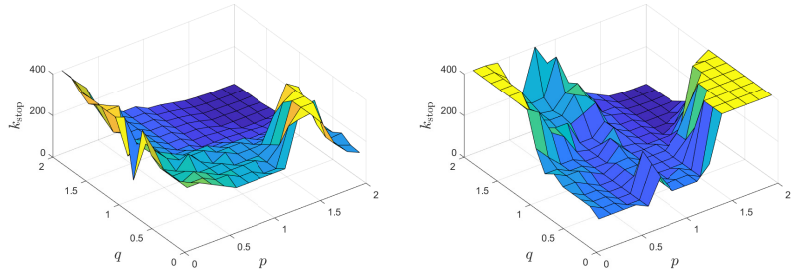


Fig. 6: Cameraman test case: RRE obtained with A-MM-GKS Restarted and μ determined by the GCV for several values of p and q.

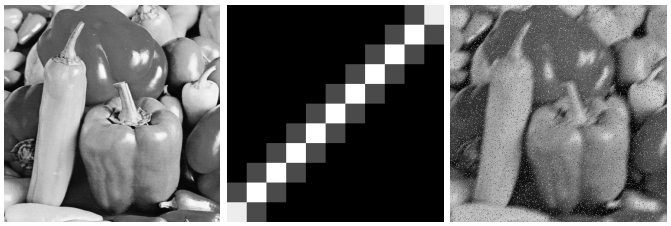


(a) (b)

Fig. 7: Cameraman test case: Number of iterations required to reach convergence with $\mu = 1$ for several values of p and q: (a) A-MM-GKS-Restarted, (b) F-MM-GKS-Restarted.

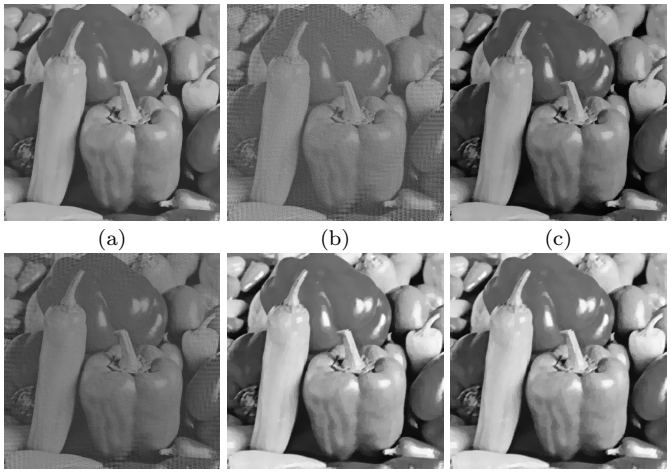
Table 2 reports results for the different restoration methods. All methods considered, except for the A-MM-GKS method paired with GCV, produce restorations of high quality. The CPU time for the standard methods are determined by the fact that the maximum number of allowed iterations is reached. Note that the restarted approaches of the present paper require significantly less computer memory than the corresponding standard approaches from [20]. The restarted adaptive MM method with the regularization parameter determined by GCV or hand-tuning yields restorations of high quality much faster than the corresponding standard method. The accuracy of the computed solutions is confirmed by visual inspection of the computed restorations shown in Fig. 9.

We would like to point out that since the GCV method is a so-called heuristic method for determining the regularization parameter, it may fail



(a) (b) (c)

Fig. 8: Peppers test case: (a) True image (500×500 pixels), (b) PSF (11×11 pixels), (c) Blurred and noisy image with 10% of impulse noise and 1% of white Gaussian noise (500×500 pixels).



(a) (b) (c)

(d) (e) (f)

Fig. 9: Peppers test case reconstructions: (a) A-MM-GKS (hand-tuned parameter), (b) A-MM-GKS (GCV parameter), (c) A-MM-GKS Restarted (hand-tuned parameter), (d) A-MM-GKS Restarted (GCV parameter), (e) F-MM-GKS (hand-tuned parameter), (f) F-MM-GKS Restarted (hand-tuned parameter).

for some problems; see [13, 22] for discussions. However, typically the method performs well.

Comparison with FISTA. We now compare the A-MM-GKS and F-MM-GKS methods to FISTA [1], and consider a Computer Tomography (CT) problem. The matrix A is a discretization of the Radon transform. An unknown two-dimensional object, formed of different materials with different attenuation coefficients, is shined on with parallel X-rays at different angles. The intensities of the X-rays are measured after traveling through the a phantom. The difference between the intensity at the source and the measured one is the sum

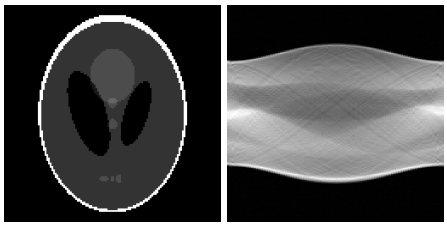
Table 2: Results for all the computed examples. For each example and considered method we report the obtained PSNR-values, the CPU time (in seconds) required for the computations, and the dimension of the search subspaces considered (for the restarted method we report the maximum dimension allowed and we write $\leq K_{\max}$).

Example	Method	Parameter	PSNR	CPU time	Dim. of subspace
Satellite	A-MM-GKS	Hand-tuned	31.070	500.00	200
		GCV	29.661	497.42	200
	A-MM-GKS Rest.	Hand-tuned	31.395	70.647	≤ 30
		GCV	30.197	70.67	≤ 30
	F-MM-GKS	Hand-tuned	32.428	18.19	112
		DP	31.819	89.94	220
	F-MM-GKS Rest.	Hand-tuned	33.278	25.85	≤ 30
		DP	31.057	14.01	≤ 30
Cameraman	A-MM-GKS	Hand-tuned	27.208	103.93	200
		GCV	27.174	92.36	200
	A-MM-GKS Rest.	Hand-tuned	27.446	13.38	≤ 30
		GCV	25.491	18.182	≤ 30
	F-MM-GKS	Hand-tuned	26.433	17.85	200
	F-MM-GKS Rest.	Hand-tuned	26.270	13.38	≤ 30
Peppers	A-MM-GKS	Hand-tuned	29.301	535.97	200
		GCV	20.863	528.17	200
	A-MM-GKS Rest.	Hand-tuned	29.420	57.73	≤ 30
		GCV	24.95	74.64	≤ 30
	F-MM-GKS	Hand-tuned	30.198	41.03	200
	F-MM-GKS Rest.	Hand-tuned	30.113	25.48	≤ 30

of the attenuation coefficients along each X-ray; see, e.g., [25] for more details on CT. To construct the operator A, we use the Matlab toolbox IR Tools [15], using 90 equispaced angles between 0 and π and shining 181 rays per angle. We assume the noise affecting the data to be white and Gaussian and the phantom to be sparse. We add 0.05% of white Gaussian noise. Therefore, in (2) we set $p = 2$ and $L = I$, where I denotes the identity matrix of appropriate size. If we set $q = 1$, then we can apply FISTA to solve the minimization problem. Figure 10 dispalys the phantom and the sinogram.

Since FISTA does not provide a way to determine the regularization parameter, we compute the solution for 10 logaritmicaly spaced values of μ between 10^{-3} and 1. Since we observed that FISTA rarely reached convergence within 200 iterations, we set the maximum number of iterations for FISTA to 2000. FISTA is compared with both A-MM-GKS Restarted and F-MM-GKS Restarted for the same fixed values of μ, and we also consider the DP and GCV criteria. Note that, since the solution in this case is scaled to be between 0 and 1, we fix $\varepsilon = 10^{-2}$ in the MM methods.

Our results are reported in Table 3. We can observe that in terms of accuracy, the methods perform similarly, which is to be expected since they are minimizing the same functional. However, we can see that the MM methods converge in fewer iterations than FISTA, especially with the fixed approach.



(a) (b)

Fig. 10: Comparison with FISTA: (a) Phantom (128×128 pixels), (b) Sinogram with 0.05% of white Gaussian noise (181×90 pixels).

(a) (b) (c)

Fig. 11: Reconstructions for the comparison with FISTA: (a) FISTA (optimal parameter), (b) A-MM-GKS Restarted (GCV parameter), (c) F-MM-GKS Restarted (DP parameter).

Moreover, generally the MM methods yield more accurate reconstructions than FISTA. The adaptive approach requires roughly 5 times the CPU time than FISTA, while the fixed approach is comparable in term of overall computational cost. This seems to suggest that FISTA is more convenient to use than the MM methods, however, firstly the MM methods allow $p \neq 1$ and can easily include a regularization matrix $L \neq I$. Moreover, if one needs to select the regularization parameter μ, then this requires to run FISTA several times and then apply some a-posteriori criterion to determine μ. The MM methods allow inexpensive determination of μ during the iterations and require just one run of the algorithm. Therefore, if we compare a single run of either A-MM-GKS Restarted with the GCV or F-MM-GKS Restarted with the DP to FISTA, then we can observe that it is computationally cheaper to run either one of the MM methods once than running FISTA several times with different values of the regularization parameter. We would like to mention that FISTA requires a fairly accurate estimate of $\|A\|_2$. This is done by the `normest` command in Matlab and we do not include this in the CPU time for FISTA, as we assume that this is performed off line. Finally, we report some reconstructions in Fig. 11. Visual inspection of these computed solutions confirms that the MM methods yield more accurate reconstructions than FISTA.

Table 3: Comparison with FISTA. For each considered method we report the obtained PSNR-values,the number of iterations performed, and the CPU time (in seconds) required for the computations. We consider 10 logarithmicaly spaced values of μ between 10^{-3} and 1 and the DP and GCV criteria for the determination of the regularization parameter.

Param.	FISTA			A-MM-GKS Rest.			F-MM-GKS Rest.		
	PSNR	Iter.	CPU time	PSNR	Iter.	CPU time	PSNR	Iter.	CPU time
μ_1	22.529	2000	3.998	26.499	200	10.145	26.827	151	6.946
μ_2	23.643	2000	4.088	26.675	200	10.110	27.093	91	4.280
μ_3	24.640	2000	4.023	27.052	200	10.163	27.051	61	2.873
μ_4	25.341	2000	4.034	27.837	200	10.170	27.219	61	2.841
μ_5	26.745	1749	3.529	29.371	200	10.297	27.536	61	2.873
μ_6	28.283	1458	2.996	31.593	200	10.109	27.716	82	3.827
μ_7	29.287	1274	2.558	33.323	200	10.126	27.513	92	4.268
μ_8	30.048	1105	2.265	34.085	200	10.228	27.765	49	2.324
μ_9	30.176	1030	2.098	34.053	200	10.212	29.118	36	1.736
μ_{10}	29.283	1072	2.171	33.501	181	9.362	30.354	32	1.544
DP	–	–	–	–	–	–	34.367	61	3.269
GCV	–	–	–	34.688	200	10.331	–	–	–

5 Conclusions

This paper describes modifications of the Majorization-Minimization algorithms proposed in [20] for the solution of minimization problems of the form (2). The algorithms in [20] may require many iterations to satisfy a user-supplied stopping criterion. Since the dimension of the solution subspace of these algorithms grows linearly with the number of iterations, the storage requirement of these algorithms may be substantial for large-scale problems. Moreover, the computational effort required increases linearly or quadratically with the number of iterations. This can make the algorithms expensive to use in terms of CPU time. This paper describes an approach to reduce the memory requirement and the computational effort required by the algorithms in [20]. Our approach restarts the algorithm with a new solution subspace every $K_{\max}$ iterations. This secures that the dimension of the solution subspace is bounded independently of the number of iterations carried out. Numerical examples illustrate that Algorithms 1 and 2 are able to produce high-quality restorations, and they require less memory and computational time than their counterparts in [20]. A new convergence proof, that holds for the algorithms of the present paper, and improves the converge proof in [20] also is supplied.

Acknowledgments

The authors would like to thank the referees for comments. A.B. is a member of the GNCS-INdAM group which partially supported his research with the project “Regularization Methods and Models for large scale inverse ill-posed problems”. Moreover, his research is partially supported by the Regione Au-

tonoma della Sardegna research project "Algorithms and Models for Imaging Science [AMIS]" (RASSR57257, intervento finanziato con risorse FSC 2014-2020 - Patto per lo Sviluppo della Regione Sardegna).

References

1. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM journal on imaging sciences **2**(1), 183–202 (2009)
2. Belge, M., Kilmer, M.E., Miller, E.L.: Efficient determination of multiple regularization parameters in a generalized L-curve. Inverse Problems **18**, 1161–1183 (2002)
3. Buccini, A., De la Cruz Cabrera, O., Donatelli, M., Martinelli, A., Reichel, L.: Large-scale regression with non-convex loss and penalty. Applied Numerical Mathematics **157**, 590–601 (2020)
4. Buccini, A., De la Cruz Cabrera, O., Koukouvinos, C., Mitrouli, M., Reichel, L.: Variable selection in saturated and supersaturated designs via minimization. Communications in Statistics - Simulation and Computation, in press (2021)
5. Buccini, A., Donatelli, M., Ramlau, R.: A semiblind regularization algorithm for inverse problems with application to image deblurring. SIAM Journal on Scientific Computing **40**(1), A452–A483 (2018)
6. Buccini, A., Pasha, M., Reichel, L.: Modulus-based iterative methods for constrained ℓ^p-ℓ^q minimization. Inverse Problems **36**(8), 084,001 (2020)
7. Buccini, A., Pragliola, M., Reichel, L., Sgallari, F.: A comparison of parameter choice rules for ℓ_p-ℓ_q minimization. Annali dell' Università di Ferrara **68**, 441–463 (2022)
8. Buccini, A., Reichel, L.: An ℓ^2-ℓ^q regularization method for large discrete ill-posed problems. Journal of Scientific Computing **78**, 1526–1549 (2019)
9. Buccini, A., Reichel, L.: Generalized cross validation for ℓ^p-ℓ^q minimization. Numerical Algorithms **88**, 1595–1616 (2021)
10. Chan, R.H., Liang, H.X.: Half-quadratic algorithm for ℓ_p-ℓ_q problems with applications to TV-ℓ_1 image restoration and compressive sensing. In: Efficient Algorithms for Global Optimization Methods in Computer Vision, Lecture Notes in Computer Science # 8293, pp. 78–103. Springer, Berlin (2014)
11. Daniel, J.W., Gragg, W.B., Kaufman, L., Stewart, G.W.: Reorthogonalization and stable algorithms for updating the Gram–Schmidt QR factorization. Mathematics of Computation **30**, 772–795 (1976)
12. Donatelli, M., Reichel, L.: Square smoothing regularization matrices with accurate boundary conditions. Journal of Computational and Applied Mathematics **272**, 334–349 (2014)
13. Engl, H.W., Hanke, M., Neubauer, A.: Regularization of Inverse Problems. Kluwer, Dordrecht (1996)
14. Fornasier, M., Naumova, V., Pereverzyev, S.V.: Parameter choice strategies for multi-penalty regularization. SIAM Journal on Numerical Analysis **52**, 1770–1794 (2014)
15. Gazzola, S., Hansen, P.C., Nagy, J.G.: IR Tools: a MATLAB package of iterative regularization methods and large-scale test problems. Numerical Algorithms **81**(3), 773–811 (2019)
16. Gazzola, S., Reichel, L.: A new framework for multi-parameter regularization. BIT Numerical Mathematics **56**, 919–949 (2016)
17. Golub, G.H., Van Loan, C.F.: Matrix Computations, 4th. edition. Johns Hopkins University Press, Baltimore (2013)
18. Hansen, P.C.: Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion. SIAM, Philadelphia (1998)
19. Hansen, P.C., Nagy, J.G., O'Leary, D.P.: Deblurring Images: Matrices, Spectra, and Filtering. SIAM, Philadelphia (2006)
20. Huang, G., Lanza, A., Morigi, S., Reichel, L., Sgallari, F.: Majorization-minimization generalized Krylov subspace methods for ℓ_p-ℓ_q optimization applied to image restoration. BIT Numerical Mathematics **57**, 351–378 (2017)

21. Ito, K., Jin, B., Takeuchi, T.: Multi-parameter Tikhonov regularization. Methods and Applications of Analysis **18**, 31–46 (2011)
22. Kindermann, S.: Convergence analysis of minimization-based noise level-free parameter choice rules for linear ill-posed problems. Electron. Trans. Numer. Anal. **38**, 233–257 (2011)
23. Kindermann, S., Raik, K.: A simplified L-curve as error estimator. Electron. Trans. Numer. Anal. **53**, 213–238 (2020)
24. Lanza, A., Morigi, S., Reichel, L., Sgallari, F.: A generalized Krylov subspace method for ℓ_p-ℓ_q minimization. SIAM Journal on Scientific Computing **37**, S30–S50 (2015)
25. Natterer, F.: The Mathematics of Computerized Tomorgraphy. SIAM, Philadelphia (2001)
26. Reichel, L., Rodriguez, G.: Old and new parameter choice rules for discrete ill-posed problems. Numerical Algorithms **63**, 65–87 (2013)
27. Rodríguez, P., Wohlberg, B.: Efficient minimization method for a generalized total variation functional. IEEE Transactions on Image Processing **18**(2), 322–332 (2009)
28. Saad, Y.: Iterative Methods for Sparse Linear Systems, 2nd. ed. SIAM, Philadelphia (2003)