Restoration of Blurred Images Corrupted by Impulse Noise via Median Filters and ℓ^p-ℓ^q Minimization

Majed Alotaibi
Department of Mathematical Sciences
Kent State University
Kent, Ohio, USA
malota15@kent.edu

Alessandro Buccini
Department of Mathematics and Computer Science
University of Cagliari
Cagliari, Italy
alessandro.buccini@unica.it

Lothar Reichel
Department of Mathematical Sciences
Kent State University
Kent, Ohio, USA
reichel@math.kent.edu

***Abstract*—This paper proposes a two-phase fully automatic scheme for restoring images that have been corrupted by blur, impulse noise, and, possibly, Gaussian noise. In a first phase, a median filter is used to partially correct most of the pixels that are contaminated by impulse noise. Then the image is restored by solving the ℓ^p-ℓ^q minimization. This problem consists of a fidelity term, that is defined in terms of a p-norm, and a regularization term, that is defined in terms of a q-norm. We allow $0 < p, q \leq 2$. In particular, the p-norm is not a norm when $0 < p < 1$; similarly for the q-norm. The relative influence of the fidelity and regularization terms is determined by a regularization parameter. We describe how this parameter can be chosen without user interaction. The numerical scheme described in this paper can restore images that have been contaminated by blur and with up to 70% of the pixels contaminated by salt-and-pepper noise.**

***Index Terms*—ℓ^p-ℓ^q minimization, modified cross validation, generalized cross validation, impulse noise, median filter.**

I. Introduction

This paper is concerned with the restoration of images that have been contaminated by blur and impulse noise possibly along with Gaussian noise. We consider problems of the form

$$\min_{x\in\mathbb{R}^n} \|Ax - b^\delta\|_2^2, \tag{1}$$

where $\|\cdot\|_2$ denotes the Euclidean norm, $A \in \mathbb{R}^{m\times n}$ is a severely ill-conditioned matrix, i.e., a matrix whose singular values decay rapidly and with no significant gap, $b^\delta \in \mathbb{R}^m$ collects the measured data that we assume is affected by noise and blur, and $x \in \mathbb{R}^n$ is the unknown image that we wish to recover. We are interested in image deblurring problems where (1) is the discretization of a Fredholm integral equation of the first kind

$$g(u,v) = \int_{\mathbb{R}^2} K(s,t,u,v) f(u,v) du dv,$$

where K is a smooth integral kernel with compact support and g and f are the continuous version of b, i.e., the noise-free data in (1), and x, respectively. The kernel K is often referred to as a Point Spread Function (PSF) and describes how a single point is spread across its neighborhood. If we assume that the blur is spatially invariant, i.e., that it does not depend on the location of the point, the integral equation above is reduced to a convolution. In this case, and if the entries of x and b^δ are ordered in a lexicographical way, the matrix A can be written as

$$A = T + E + R,$$

where T is a block Toeplitz with Toeplitz blocks matrix, E has small norm, and R is of small rank. The matrices E and R collect the imposed boundary conditions. Here we stress that, thanks to its structure, matrix-vector products with A can be performed cheaply; see [14] for more details on image deblurring.

Due to the ill-conditioning of A and the presence of noise in b^δ, the naïve solution $A^\dagger b^\delta$ of (1), where $A^\dagger$ denotes the Moore–Penrose pseudo-inverse of A, usually does not provide a meaningful approximation of the true image $x_{\text{true}} = A^\dagger b$. Here the vector $b := Ax_{\text{true}}$ represents the unavailable blurred but noise-free image associated with x_{true}. We assume that the noise that corrupts the data is a mixture of impulsive, salt-and-pepper, and Gaussian noise; see below for more details.

To recover an approximation of x_{true}, we proceed as follows. We preprocess the data to remove the impulse and salt-and-pepper noise by applying a median filter to b^δ. This

A.B. is a member of the GNCS group of INdAM. A.B. is partially supported by the Regione Autonoma 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). Research by L.R. is supported in part by NSF grant DMS-1720259.

gives the vector $\widehat{b}^\delta$. Then to deblur the image, we solve the minimization problem

$$\min_{x\in\mathbb{R}^n} \mathcal{J}(x), \quad \mathcal{J}(x) := \frac{1}{p}\|Ax - \widehat{b}^\delta\|_p^p + \frac{\mu}{q}\|Lx\|_q^q, \quad (2)$$

where $0 < p, q \leq 2$, the map $x \mapsto \|x\|_p^p$ is defined by

$$\|x\|_p^p = \sum_{j=1}^{n} |x_j|^p, \quad x = [x_1, x_2, \ldots, x_n]^T,$$

and $L \in \mathbb{R}^{r\times n}$ is a regularization matrix. Note that, if $p \geq 1$, then $\|x\|_p$ is a norm. The choice of the parameters $0 < p, q \leq 2$ affects the quality of the computed restoration as well as the properties of the functional $\mathcal{J}(x)$. This functional is convex and smooth when $1 < p, q \leq 2$, and non-convex and non-smooth when $0 < p < 1$ or $0 < q < 1$. We discuss the choices of p and q below. Minimization problems of the form (2) have been discussed extensively in the literature, see, e.g., [1]–[5], [8], [10].

The first term in the expression for $\mathcal{J}(x)$ often is referred to as the *fidelity term*, whereas the second term is called the *regularization term*. The purpose of the regularization term is to avoid a solution of (2) that is severely contaminated by propagated and amplified noise; the regularization parameter $\mu > 0$ determines the relative importance of the fidelity and regularization terms. This parameter will be determined by the numerical method presented.

The choice of $0 < p \leq 2$ in the fidelity term should depend on the type of noise that contaminates b^δ. For Gaussian noise, the value $p = 2$ is appropriate, while for impulse noise, $p \leq 1$ yields good results; see [2], [19] for discussions and illustrations. The choice of $0 < q \leq 2$ in the regularization term determines the sparsity of the computed solution; the smaller the q-value, the better the computed solution approximates a sparse solution, i.e., a solution with many vanishing entries. The value $q = 1$ is commonly used, because it provides a sparse solution and makes the functional (2) convex if $p \geq 1$. However, solutions of higher quality often can be determined when $0 < q < 1$; see [4], [15] for illustrations.

We will solve the minimization problem (2) using algorithms described in [15]. These algorithms consist of two stages. First, one determines a quadratic surrogate functional $x \mapsto \mathcal{Q}(x, x^{(k)})$ that locally approximates the functional $\mathcal{J}(x)$ at the currently available approximation $x^{(k)}$ of the minimum of $\mathcal{J}(x)$; the functional $\mathcal{Q}(x, x^{(k)})$ is chosen to be a quadratic tangent majorant of $\mathcal{J}(x)$ at $x^{(k)}$; see Section II for more details. Then $x \mapsto \mathcal{Q}(x, x^{(k)})$ is minimized in a generalized Krylov subspace. This furnishes an improved approximation, $x^{(k+1)}$, of the minimizer of $\mathcal{J}(x)$. We refer to this kind of minimization schemes for (2) as majorization-minimization generalized Krylov subspace (MM-GKS) methods; details and convergence results can be found in [15].

The choice of the regularization parameter $\mu > 0$ is important for the quality of the computed restoration. A too large value of μ may result in unnecessary lack of details in the restored image, while a too small value of $\mu > 0$ may lead to significant propagation of the noise in $\widehat{b}^\delta$ into the computed restoration. We consider two automatic strategies for determining the regularization based on cross validation (CV) and generalized cross validation (GCV). They have previously been presented in [3], [5]; here we apply them in a different way.

The data $\widehat{b}^\delta$ for the minimization problem (2) is obtained by applying a median filter to b^δ. We will use the median filters proposed in [9], [16]. These filters detect and replace the pixels that have been corrupted by impulse noise by a weighted median of neighboring pixel values. The application of median filters may produce images with blurred edges. However, since the given images also are corrupted by blur, we apply an ℓ^p-ℓ^q minimization method outlined above to determine a better approximation of x_{true} than $\widehat{b}^\delta$. This is similar to the approach of outlier smoothing followed by deblurring described by Cai et al. [6]. We note that the regularization parameter in the methods presented in [6] is tuned by the user. In our method, this parameter is chosen without user interaction. Sciacchitano et al. [20] propose a two-phase method that does not require a user to specify a regularization parameter, but this method is not designed to remove mixed noise, i.e., noise that is made up of both impulse noise and Gaussian noise. Our image restoration methods consist of the following two steps:

1) Noise reduction of the given data b^δ by application of a median filter.
2) Edge-preserving restoration by solving the minimization problem (2) with an automatic selection of the regularization parameter.

These steps define fully automatic image restoration methods that can handle images with more impulse noise combined with Gaussian noise than available automatic restoration methods, such as those described in [3], [5].

This paper is organized as follows: Section II reviews the majorization and minimization steps of the MM-GKS methods that are applied to the minimization of (2), discusses how to determine the regularization parameter μ in (2), describes the median filters used, and, finally, presents the proposed schemes. Computed examples are reported in Section III, and concluding remarks can be found in Section IV.

II. The proposed approaches

We describe the details of our algorithms. First, we discuss the majorization-minimization approaches described in [15]. Then we report two automatic ways to determine the regularization parameter proposed in [3], [5]. This is followed by a discussion on two median filters [7], [9]. Finally, we describe our methods.

A. Majorization-minimization

1) Majorization: The majorization steps considered have been proposed in [15]. At the kth step, we seek to find a quadratic tangent majorant $\mathcal{Q}(x, x^{(k)})$ of $\mathcal{J}(x)$ at the point $x^{(k)}$.

Definition 1. *Let $\mathcal{J}(x) : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function. The function $\mathcal{Q}(x, x^{(k)}) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is*

said to be a quadratic tangent majorant for $\mathcal{J}(x)$ if and only if for any $y \in \mathbb{R}^n$ all the following conditions hold:

- *$\mathcal{Q}(x, y)$ is quadratic;*
- *$\mathcal{Q}(x, y) \geq \mathcal{J}(x)$ for all $x \in \mathbb{R}^n$;*
- *$\mathcal{Q}(y, y) = \mathcal{J}(y)$ and $\nabla_x \mathcal{Q}(y, y) = \nabla_x \mathcal{J}(y)$, where ∇_x denotes the gradient with respect to the variable x.*

Definition 1 asserts that both the majorant and the objective function must have the same gradient at the current point $x^{(k)}$. Since $\mathcal{J}(x)$ defined in (2) is not smooth for $0 < p, q \leq 1$, it is common to smooth $\mathcal{J}(x)$ by introducing a smoothed "norms" as follows: Define

$$\Phi_{z,\varepsilon}(t) := \left(\sqrt{t^2 + \varepsilon^2}\right)^z \quad \text{with} \quad \begin{cases} \varepsilon > 0 & \text{if } z \in (0, 1], \\ \varepsilon = 0 & \text{if } z \in (1, 2]. \end{cases}$$

We then replace the non-smooth ℓ^p-ℓ^q minimization problem (2) (for our values of p and q) by a smoothed minimization problem

$$\mathcal{J}_\varepsilon(x) := \frac{1}{p} \sum_{i=1}^{m} \Phi_{p,\varepsilon}\left(\left(Ax - \widehat{b}^\delta\right)_i\right) + \frac{\mu}{q} \sum_{i=1}^{r} \Phi_{q,\varepsilon}\left((Lx)_i\right),$$

Huang et al. [15] discussed the construction of fixed or adaptive majorants. The fixed majorant is a quadratic functional, whose aperture does not depend on $x^{(k)}$ and which is translated so that it is tangent to $\mathcal{J}_\varepsilon$ at $x^{(k)}$. The adaptive majorant is the quadratic tangent majorant of $\mathcal{J}_\varepsilon$ at $x^{(k)}$ that has the widest aperture possible.

In the fixed case, a majorant with a fixed aperture is constructed and translated at each iteration step. It has the appealing computational advantage over using adaptive majorants that the required QR factorizations, which are used when solving certain least squares problems defined below, can be updated in each step; the use of adaptive majorants requires that the QR factorizations be recomputed from scratch in every iteration step. Let

$$\begin{aligned} v^{(k)} &:= Ax^{(k)} - \widehat{b}^\delta, \\ u^{(k)} &:= Lx^{(k)}. \end{aligned}$$

The *fixed majorant* $\mathcal{Q}^F(x, x^{(k)})$ of $\mathcal{J}_\varepsilon(x)$ at the point $x^{(k)}$ is given by

$$\begin{aligned} \mathcal{Q}^F\left(x, x^{(k)}\right) = &\frac{\varepsilon^{p-2}}{2}\left(\|Ax - \widehat{b}^\delta\|_2^2 - 2\left\langle \omega_{\text{fid}}^{(k)}, Ax \right\rangle\right) \\ &+ \frac{\mu\varepsilon^{q-2}}{2}\left(\|Lx\|_2^2 - 2\left\langle \omega_{\text{reg}}^{(k)}, Lx \right\rangle\right) + c, \end{aligned} \tag{3}$$

where c is a constant that is obtained by collecting all terms that are independent of x, and

$$\begin{aligned} \omega_{\text{fid}}^{(k)} &= v^{(k)}\left(1 - \left(\frac{\left(v^{(k)}\right)^2 + \varepsilon^2}{\varepsilon^2}\right)^{p/2-1}\right), \\ \omega_{\text{reg}}^{(k)} &= u^{(k)}\left(1 - \left(\frac{\left(u^{(k)}\right)^2 + \varepsilon^2}{\varepsilon^2}\right)^{q/2-1}\right), \end{aligned}$$

where all operations are component-wise.

For *adaptive majorization*, the quadratic majorant is constructed so that it has the maximal aperture possible at each iteration step. The adaptive majorant $\mathcal{Q}^A(x, x^{(k)})$ of $\mathcal{J}_\varepsilon(x)$ at the point $x^{(k)}$ is given by

$$\begin{aligned} \mathcal{Q}^A\left(x, x^{(k)}\right) = &\frac{1}{2}\left\|\left(W_{\text{fid}}^{(k)}\right)^{1/2}\left(Ax - \widehat{b}^\delta\right)\right\|_2^2 \\ &+ \frac{\mu}{2}\left\|\left(W_{\text{reg}}^{(k)}\right)^{1/2} Lx\right\|_2^2 + c, \end{aligned} \tag{4}$$

where

$$\begin{aligned} W_{\text{fid}}^{(k)} &= \text{diag}\left(\left(v^{(k)}\right)^2 + \varepsilon^2\right)^{p/2-1}, \\ W_{\text{reg}}^{(k)} &= \text{diag}\left(\left(u^{(k)}\right)^2 + \varepsilon^2\right)^{q/2-1}, \end{aligned}$$

and c is a constant independent of x.

2) Minimization: Minimizing (4) and (3) over x can be done by solving the normal equations associated with the fixed and adaptive MM-GKS methods

$$\left(A^T A + \eta L^T L\right) x = A^T(\widehat{b}^\delta + \omega_{\text{fid}}^{(k)}) + \eta L^T \omega_{\text{reg}}^{(k)}, \tag{5}$$

$$\left(A^T W_{\text{fid}}^{(k)} A + \mu L^T W_{\text{reg}}^{(k)} L\right) x = A^T W_{\text{fid}}^{(k)} \widehat{b}^\delta, \tag{6}$$

respectively, where

$$\eta := \mu \frac{\varepsilon^{q-2}}{\varepsilon^{p-2}}.$$

We remark that for numerical reasons, we compute the solutions of (5) or (6) without explicitly forming the normal equations; see below. The linear systems of equations (5) and (6) have unique solutions if only if, for the fixed case

$$\mathcal{N}\left(A^T A\right) \cap \mathcal{N}\left(L^T L\right) = \{0\},$$

and for the adaptive case

$$\mathcal{N}\left(A^T W_{\text{fid}}^{(k)} A\right) \cap \mathcal{N}\left(L^T W_{\text{reg}}^{(k)} L\right) = \{0\},$$

for all k, where $\mathcal{N}(M)$ denotes the null space of the matrix M. These conditions are satisfied in image restoration by the definition of A and L and by the fact that $W_{\text{fid}}^{(k)}$ and $W_{\text{reg}}^{(k)}$ are positive definite matrices.

a) The Fixed Case: We seek a solution of (5) in a low-dimensional search space. Let $V_k \in \mathbb{R}^{n \times m_k}$ with $0 < m_k \ll n$, where the columns of V_k form an orthonormal basis for the search space. By letting

$$x^{(k+1)} = V_k y^{(k+1)}, \tag{7}$$

we find $y^{(k+1)}$ by solving the following minimization problem

$$y^{(k+1)} := \arg\min_{y \in \mathbb{R}^{m_k}} \left\| \begin{bmatrix} AV_k \\ \eta^{1/2} LV_k \end{bmatrix} y - \begin{bmatrix} \widehat{b}^\delta + \omega_{\text{fid}}^{(k)} \\ \eta^{1/2} \omega_{\text{reg}}^{(k)} \end{bmatrix} \right\|_2^2. \tag{8}$$

Let us consider the following QR factorizations

$$\begin{aligned} AV_k &= Q_A R_A \quad \text{with} \quad Q_A \in \mathbb{R}^{m \times m_k}, \quad R_A \in \mathbb{R}^{m_k \times m_k}, \\ LV_k &= Q_L R_L \quad \text{with} \quad Q_L \in \mathbb{R}^{r \times m_k}, \quad R_L \in \mathbb{R}^{m_k \times m_k}, \end{aligned} \tag{9}$$

where Q_A and Q_L have orthonormal columns and R_A and R_L are upper-triangular matrices. By substituting (9) into (8), we obtain the minimization problem

$$y^{(k+1)} := \arg\min_{y\in\mathbb{R}^{m_k}} \left\| \begin{bmatrix} R_A \\ \eta^{1/2} R_L \end{bmatrix} y - \begin{bmatrix} Q_A^T \left(\widehat{b}^\delta + \omega_{\text{fid}}^{(k)} \right) \\ \eta^{1/2} Q_L^T \omega_{\text{reg}}^{(k)} \end{bmatrix} \right\|_2^2$$

with associated normal equations

$$\left(R_A^T R_A + \eta R_L^T R_L \right) y^{(k+1)} = R_A^T Q_A^T \left(\widehat{b}^\delta + \omega_{\text{fid}}^{(k)} \right) + \eta R_L^T Q_L^T \omega_{\text{reg}}^{(k)}.$$

Substituting (7) into (5) gives the residual vector

$$r^{(k+1)} := A^T \left(A V_k y^{(k+1)} - \widehat{b}^\delta - \omega_{\text{fid}}^{(k)} \right) + \eta L^T \left(L V_k y^{(k+1)} - \omega_{\text{reg}}^{(k)} \right).$$

As suggested by Lampe et al. [18], we start with the initial solution subspace V_1. Then V_k is expanded to V_{k+1} by adding $v_{\text{new}} := r^{(k+1)}/\|r^{(k+1)}\|_2$, i.e., $V_{k+1} = [V_k, v_{\text{new}}]$ for $k = 1, 2, \ldots$. These solution subspaces are referred to as a generalized Krylov subspace. The space V_1 can be determined, for instance, by a few steps of Golub–Kahan bidiagonalization applied to A with initial vector $\widehat{b}^\delta$; see, e.g., [15], [18]. In our experiments, we let $V_1 = A^T\widehat{b}^\delta / \left\| A^T \widehat{b}^\delta \right\|_2$.

The QR factorizations in (9) are updated according to

$$AV_{k+1} = [AV_k, Av_{\text{new}}] = [Q_A, \tilde{q}_A] \begin{bmatrix} R_A & r_A \\ 0^T & \tau_A \end{bmatrix},$$
$$LV_{k+1} = [LV_k, Lv_{\text{new}}] = [Q_L, \tilde{q}_L] \begin{bmatrix} R_L & r_L \\ 0^T & \tau_L \end{bmatrix},$$

where

$$\begin{aligned} r_A &= Q_A^T (A v_{\text{new}}), & q_A &= A v_{\text{new}} - Q_A r_A, \\ \tau_A &= \|q_A\|_2, & \tilde{q}_A &= q_A/\tau_A, \\ r_L &= Q_L^T (L v_{\text{new}}), & q_L &= L v_{\text{new}} - Q_L r_L, \\ \tau_L &= \|q_L\|_2, & \tilde{q}_L &= q_L/\tau_L. \end{aligned}$$

b) The Adaptive Case: Similarly to the fixed case, we determine an approximate solution $x^{(k+1)}$ of (6) restricted to the column space of V_k by determining the solution $y^{(k+1)}$ of the minimizing problem

$$\min_{y\in\mathbb{R}^{m_k}} \left\| \begin{bmatrix} \left(W_{\text{fid}}^{(k)}\right)^{1/2} A V_k \\ \mu^{1/2} \left(W_{\text{reg}}^{(k)}\right)^{1/2} L V_k \end{bmatrix} y - \begin{bmatrix} \left(W_{\text{fid}}^{(k)}\right)^{1/2} \widehat{b}^\delta \\ 0 \end{bmatrix} \right\|_2^2 \quad (10)$$

and then computing $x^{(k+1)} = V_k y^{(k+1)}$. To this end, we calculate the QR factorizations

$$\begin{aligned} &\left(W_{\text{fid}}^{(k)}\right)^{1/2} A V_k = Q_A R_A \text{ with } Q_A \in \mathbb{R}^{m\times m_k},\ R_A \in \mathbb{R}^{m_k\times m_k}, \\ &\left(W_{\text{reg}}^{(k)}\right)^{1/2} L V_k = Q_L R_L \text{ with } Q_L \in \mathbb{R}^{r\times m_k},\ R_L \in \mathbb{R}^{m_k\times m_k}. \end{aligned} \quad (11)$$

Substituting (11) into (10) yields the minimization problem

$$\min_{y\in\mathbb{R}^{m_k}} \left\| \begin{bmatrix} R_A \\ \mu^{1/2} R_L \end{bmatrix} y - \begin{bmatrix} Q_A^T W_{\text{fid}}^{1/2} \widehat{b}^\delta \\ 0 \end{bmatrix} \right\|_2^2,$$

which we solve for $y^{(k+1)} = y$. This solution satisfies the normal equations

$$\left(R_A^T R_A + \mu R_L^T R_L \right) y^{(k+1)} = R_A^T Q_A^T \left(W_{\text{fid}}^{(k)} \right)^{1/2} \widehat{b}^\delta. \quad (12)$$

The residual of the normal equations (6) associated with $x^{(k+1)} = V_k y^{(k+1)}$ is given by

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

Similarly to the fixed case, we start with the initial solution subspace $V_1 = A^T\widehat{b}^\delta / \left\| A^T \widehat{b}^\delta \right\|_2$. Subsequently V_k is expanded to V_{k+1} by including the vector $r^{(k+1)}/\|r^{(k+1)}\|_2$.

B. Choice of the Regularization Parameter

This section describes two methods for choosing the regularization parameter, the MCV and GCV methods. These methods have previously been described in [3], [5].

1) Modified Cross Validation: The classical Cross Validation (CV) method splits the data into two complementary sets: the training set and the testing set. These sets consist of complementary subsets of equations. The training set is used for solving the problem with different regularization parameters, and the second set is used to validate the choice of regularization parameter. Specifically, the CV method chooses the parameter μ such that the solution determined by using the training set provides a small residual error when substituted into the data of the testing set.

The Modified Cross Validation (MCV) method uses a modified version of the CV approach. Instead of seeking to reconstruct entries of the data vector $\widehat{b}^\delta$, the method applies CV to the computed solutions. Thus, MCV compares predictions of computed solutions for different parameters μ_j. Roughly, the MCV method determines the parameter μ that is least sensitive to loss of data. The MCV method usually gives approximations of the desired solution of higher quality than the CV method; see [5].

Consider solving problem (2) with an appropriate regularization parameter choice. Let I_1 and I_2 be two distinct sets of d distinct random integers between 1 and n. Let $\widetilde{A}_i$ and $\widetilde{b}_i^\delta$ be defined by removing the rows with index in I_i from A and $\widehat{b}^\delta$, respectively, for $i \in \{1, 2\}$. This gives the matrix $\widetilde{A}_i \in \mathbb{R}^{\widetilde{m}\times n}$ and vector $\widetilde{b}_i^\delta \in \mathbb{R}^{\widetilde{m}}$, where $\widetilde{m} < m$, for $i \in \{1, 2\}$. Let $\{\mu_j\}_{j=1}^l$ be a set of positive regularization parameters, and let $x_{\mu_j}^{(i)}$ be defined by

$$x_{\mu_j}^{(i)} := \arg\min_{x\in\mathbb{R}^n} \left\{ \frac{1}{p} \sum_{s=1}^{\widetilde{m}} \Phi_{p,\varepsilon} \left(\left(\widetilde{A}_i x - \widetilde{b}_i^\delta \right)_s \right) + \frac{\mu_j}{q} \sum_{s=1}^{r} \Phi_{q,\varepsilon} \left((Lx)_s \right) \right\}, \quad i = 1, 2.$$

Then we calculate the quantities

$$\Delta x_j = \left\| x_{\mu_j}^{(1)} - x_{\mu_j}^{(2)} \right\|_2, \quad j = 1, 2, \ldots, l.$$

This procedure is repeated K times to reduce statistical variability. Each step k provides a regularization parameter $\mu^{(k)}$, obtained by

$$\mu^{(k)} := \arg\min_{\mu_j} \Delta x_j.$$

Then the regularization parameter $\mu = \mu^*$ is chosen to be the average of the parameter values $\mu^{(k)}$, i.e.,

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

Finally, we use the determined parameter μ^* to compute

$$x^* := \arg\min_{x\in\mathbb{R}^n} \left\{ \frac{1}{p}\|Ax - \widehat{b}^\delta\|_p^p + \frac{\mu^*}{q}\|Lx\|_q^q \right\}.$$

To numerically solve all the minimization problems above, we used the fixed majorization version of MM-GKS. We could have used the adaptive version of MM-GKS instead, but the computations for this method are more demanding, since the matrices Q and R in (11) have to be computed from scratch in each iteration. Algorithm 1 summarizes the fixed majorization version of MM-GKS and Algorithm 2 describes the MCV procedure.

Algorithm 1: MM-GKS

Let $0 < p, q \leq 2$ and $\mu > 0$. Consider $A \in \mathbb{R}^{m\times n}$ and $L \in \mathbb{R}^{r\times n}$. Fix $\varepsilon > 0$ and $\eta = \mu\frac{\varepsilon^{q-2}}{\varepsilon^{p-2}}$;
Choose the initial vector $x_1 = A^T\widehat{b}^\delta$ and let $V_1 = A^T\widehat{b}^\delta / \left\|A^T\widehat{b}^\delta\right\|$;
Compute and store $AV_1 = Q_A R_A$ and $LV_1 = Q_L R_L$;
for $k = 1, 2, \ldots$ **do**
  $v^{(k)} = Ax^{(k)} - \widehat{b}^\delta$;
  $u^{(k)} = Lx^{(k)}$;
  $\omega_{\text{fid}}^{(k)} = v^{(k)}\left(1 - \left(\frac{\left(\mathrm{v}^{(k)}\right)^2+\varepsilon^2}{\varepsilon^2}\right)^{p/2-1}\right)$;
  $\omega_{\text{reg}}^{(k)} = u^{(k)}\left(1 - \left(\frac{\left(\mathrm{u}^{(k)}\right)^2+\varepsilon^2}{\varepsilon^2}\right)^{q/2-1}\right)$;
  $y^{(k+1)} = \left(R_A^T R_A + \eta R_L^T R_L\right)^{-1}\left(R_A^T Q_A^T\left(\widehat{b}^\delta + \omega_{\text{fid}}^{(k)}\right) + \eta R_L^T Q_L^T \omega_{\text{reg}}^{(k)}\right)$;
  $r = A^T\left(AV_k y^{(k+1)} - \widehat{b}^\delta - \omega_{\text{fid}}^{(k)}\right) + \eta L^T\left(LV_k y^{(k+1)} - \omega_{\text{reg}}^{(k)}\right)$;
  $v_{\text{new}} = r/\|r\|_2; V_{k+1} = [V_k, v_{\text{new}}]$;
  Update the QR factorizations $AV_{k+1} = Q_A R_A$ and $LV_{k+1} = Q_L R_L$;
  $x^{(k+1)} = V_k y^{(k+1)}$;
end

Note that in Algortihm 2 we use Algorithm 1 to compute $x_{\mu_j}^{(k,i)}$, where $x_{k,j,i}^{\text{init}} = \widetilde{A}_i^T\widetilde{b}_i^\delta$ for $j = 1, \ldots, l$, $k = 1, \ldots, K$, and $i = 1, 2$, and $V_{k,j,i}^{\text{init}} = \widetilde{A}_i^T\widetilde{b}_i^\delta / \left\|\widetilde{A}_i^T\widetilde{b}_i^\delta\right\|_2$ for $j = 1, \ldots, l$, $k = 1, \ldots, K$, and $i = 1, 2$. These choices allow us to run the various instances of the MM-GKS algorithm in parallel.

Algorithm 2: MM-GKS-MCV

Let $A \in \mathbb{R}^{m\times n}$ and $L \in \mathbb{R}^{r\times n}$. Let $d < m$, and let $K > 0$ be an integer. Let $\{\mu_j\}_{j=1}^l$ be a set of positive regularization parameters;
for $k = 1, 2, \ldots, K$ **do**
  Construct two distinct sets $I_1^{(k)}$ and $I_2^{(k)}$ of d distinct random integers between 1 and n; For $i = 1, 2$, let $\widetilde{A}_i$ and $\widetilde{b}_i^\delta$ be obtained by removing the rows with indices in $I_i^{(k)}$ from A and $\widehat{b}^\delta$, respectively;
  for $j = 1, 2, \ldots l$ **do**
    for $i = 1, 2$ **do**
      $x_{\mu_j}^{(k,i)} = \text{MM-GKS}\left(\widetilde{A}_i, L, \widetilde{b}_i^\delta, p, q, \mu_j, \varepsilon, x_{k,j,i}^{\text{init}}, V_{k,j,i}^{\text{init}}\right)$;
    end
    Compute $\Delta x_j^{(k)} = \left\|x_{\mu_j}^{(k,1)} - x_{\mu_j}^{(k,2)}\right\|_2$;
  end
  $j^* = \arg\min_{1\leq j\leq l}\left\{\Delta x_j^{(k)}\right\}$;
  $\mu^{(k)} = \mu_{j^*}$;
end
Compute $\mu = \frac{1}{K}\sum_{k=1}^K \mu^{(k)}$;

2) Generalized Cross Validation: In this scheme, μ is chosen such that the GCV function is minimized at each iteration. We use the adaptive version of MM-GKS since the normal equation (12) associated with the minimization problem are of the form

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

which is analogous to Tikhonov regularization in general form. Since R_A and R_L are constructed from projecting a large problem to low-dimensional subspace ν_k, where $\dim(\nu_k) = m_k \ll n$, we can use the GSVD for the GCV computation to determine a suitable value of the regularization parameter μ, see, e.g., [11], [12]. The GSVD of the matrix pair $\{R_A, R_L\}$ is given by

$$R_A = U\Sigma_A Y^T,$$
$$R_L = V\Sigma_L Y^T,$$

where the matrices U and V have orthonormal columns, Σ_A and Σ_L are nonnegative diagonal matrices, and Y is a non-singular matrix; see [13] for details. We define the following quantities for $\mu > 0$,

$$\begin{aligned} r_\mu^{(k+1)} &= \left\| Q_A^T\left(W_{\text{fid}}^{(k)}\right)^{1/2}\widehat{b}^\delta - R_A y^{(k+1)} \right\|_2^2 \\ &= \left\| U^T Q_A^T\left(W_{\text{fid}}^{(k)}\right)^{1/2}\widehat{b}^\delta \right. \\ &\qquad \left. -\Sigma_A^2\left(\Sigma_A^2 + \mu\Sigma_L^2\right)^{-1} U^T Q_A^T\left(W_{\text{fid}}^{(k)}\right)^{1/2}\widehat{b}^\delta \right\|_2^2 \end{aligned} \tag{13}$$

and

$$t_\mu^{(k+1)} = \left(\text{trace}\left(I - R_A\left(R_A^T R_A + \mu R_L^T R_L\right)^{-1} R_A^T\right)\right)^2 = \left(\text{trace}\left(I - \Sigma_A^2\left(\Sigma_A^2 + \mu\Sigma_L^2\right)^{-1}\right)\right)^2. \quad (14)$$

The above derivation uses equation (12) and the fact that Σ_A and Σ_L are square matrices. Then the regularization parameter determined by the GCV method is given by

$$\mu^{(k+1)} = \arg\min_\mu \frac{r_\mu^{(k+1)}}{t_\mu^{(k+1)}};$$

see [3] for a derivation. If the solution of the above minimization problem is not unique, then the largest minimizer is chosen. We summarize these computations in Algorithm 3.

Algorithm 3: MM-GKS-GCV

1 Let $0 < p, q \leq 2$. Consider $A \in \mathbb{R}^{m\times n}$ and $L \in \mathbb{R}^{r\times n}$. Fix $\varepsilon > 0$ and choose the initial vector $x_1 = A^T\widehat{b}^\delta$;
2 Let $V_1 = A^T\widehat{b}^\delta / \left\|A^T\widehat{b}^\delta\right\|$; compute and store AV_1 and LV_1 ;
3 **for** $k = 1, 2, \ldots$ **do**
4 $v^{(k)} = Ax^{(k)} - \widehat{b}^\delta$;
5 $u^{(k)} = Lx^{(k)}$;
6 $\omega_{\text{fid}}^{(k)} := \left(\left(v^{(k)}\right)^2 + \varepsilon^2\right)^{p/2-1}$;
7 $\omega_{\text{reg}}^{(k)} = \left(\left(u^{(k)}\right)^2 + \varepsilon^2\right)^{q/2-1}$;
8 $W_{\text{fid}}^{(k)} = \text{diag}\left(\omega_{\text{fid}}^{(k)}\right)$;
9 $W_{\text{reg}}^{(k)} = \text{diag}\left(\omega_{\text{reg}}^{(k)}\right)$;
10 Compute the QR factorization $Q_A R_A = \left(W_{\text{fid}}^{(k)}\right)^{1/2} AV_k$;
11 Compute the QR factorization $Q_L R_L = \left(W_{\text{reg}}^{(k)}\right)^{1/2} LV_k$;
12 Compute the GSVD of the pair $\{R_A, R_L\}$;
13 $\mu = \arg\min_\mu \frac{r_\mu^{(k+1)}}{t_\mu^{(k+1)}}$, where $r_\mu^{(k+1)}$ and $t_\mu^{(k+1)}$ are defined in (13) and (14);
14 $y^{(k+1)} = \left(R_A^T R_A + \mu R_L^T R_L\right)^{-1} R_A^T Q_A^T \left(W_{\text{fid}}^{(k)}\right)^{1/2} \widehat{b}^\delta$;
15 $r = A^T W_{\text{fid}}^{(k)}\left(AV_k y^{(k+1)} - \widehat{b}^\delta\right) + \mu L^T W_{\text{reg}}^{(k)} LV_k y^{(k+1)}$;
16 $v_{\text{new}} = r/\|r\|_2; \quad V_{k+1} = [V_k, v_{\text{new}}]$;
17 Compute and store AV_{k+1} and LV_{k+1} ;
18 $x^{(k+1)} = V_k y^{(k+1)}$;
19 **end**

3) Convergence properties: We now briefly comment on the convergence properties of the two rules for the selection of the regularization parameter. Both rules are so-called heuristic parameter choice rules and, therefore, we are not able to show regularization properties; see, e.g., [17] for a discussion of several heuristic parameter choice rules. However, for the MCV algorithm we can state that the procedure converges and provides a value for μ in a finite amount of time. On the other hand, at this time, we are not able to assure that the GCV method converges. Nevertheless, in practice the MM-GKS-GCV algorithm always converges and provides approximate solutions that are, in general, more accurate than the ones obtained with MCV.

C. Median Filters

1) Adaptive Median Filter: This section summarizes properties of the adaptive median filter (AMF) described in [16]. We use the description of the AMF method presented in [7]. Let $\widetilde{b}$ be an $r_1 \times r_2$ matrix reshape of b^δ such that $\widetilde{b}_{i,j}$ is the gray level of the noisy image at pixel (i, j), where $(i, j) \in \{1, 2, \ldots, r_1\} \times \{1, 2, \ldots, r_2\}$ and, obviously $m = r_1 r_2$. Denote by $[\widetilde{b}_{\min}, \widetilde{b}_{\max}]$ the dynamic range of $\widetilde{b}$. Let b be the unknown noise-free image that has the same size and dynamic range as $\widetilde{b}$. When $\widetilde{b}$ is corrupted by salt-and-pepper noise, it is given by

$$\widetilde{b}_{i,j} = \begin{cases} \widetilde{b}_{\min} & \text{with probability } \sigma_1, \\ \widetilde{b}_{\max} & \text{with probability } \sigma_2, \\ b_{i,j} & \text{with probability } 1 - \sigma_1 - \sigma_2, \end{cases}$$

where $\sigma = \sigma_1 + \sigma_2$ is the noise level and $b_{i,j}$ is the noise-free image at pixel (i, j). Let $B_{i,j}^w$ be a window of size $w \times w$ centered at (i, j) defined by

$$B_{i,j}^w = \{(s_1, s_2) : |s_1 - i| \leq w \text{ and } |j - s_2| \leq w\},$$

and let $w_{\max}$ denote the maximal window size. Let $\widetilde{b}_{i,j}^{w,\min}$, $\widetilde{b}_{i,j}^{w,\max}$, and $\widetilde{b}_{i,j}^{w,\text{median}}$ stand for the minimum, maximum, and median values of the pixels in $\widetilde{b}$ with index $B_{i,j}^w$, respectively.

We now describe how we denoise a pixel. Fix a starting value w and compute $\widetilde{b}_{i,j}^{w,\min}$, $\widetilde{b}_{i,j}^{w,\max}$, and $\widetilde{b}_{i,j}^{w,\text{median}}$. If

$$\widetilde{b}_{i,j}^{w,\min} < \widetilde{b}_{i,j}^{w,\text{median}} < \widetilde{b}_{i,j}^{w,\max},$$

then we assume that the pixel is not noisy; otherwise, we identify this pixel as noisy and substitute it by $\widetilde{b}_{i,j}^{w,\text{median}}$.

On the other hand, if either $\widetilde{b}_{i,j}^{w,\text{median}} = \widetilde{b}_{i,j}^{w,\min}$ or $\widetilde{b}_{i,j}^{w,\text{median}} = \widetilde{b}_{i,j}^{w,\max}$, then we enlarge the size of the window by 2, i.e., we set $w_{\text{new}} = w + 2$. If w_{new} is larger than the maximum size of the window $w_{\max}$, then we assume that the pixel $\widetilde{b}_{i,j}$ is noisy and substitute it by $\widetilde{b}_{i,j}^{w,\text{median}}$; otherwise we reiterate the procedure above with $w = w_{\text{new}}$.

Algorithm 4 summarizes these computations. The maximum window size $w_{\max}$ in Algorithm 4 depends on the salt-and-pepper noise level. As the noise level increases, the value of $w_{\max}$ should increase accordingly. However, as suggested in [7], the value of $w_{\max}$ is set to 39 if the noise level is not known. Even though this is a large value for the maximum window size, Algorithm 4 will terminate the while loop in step 5 earlier when the noise level is small than when the noise level is large. This automatically reduces the computational effort and makes AMF capable of filtering images with any noise level.

Algorithm 4: The AMF method

Consider the blurred and noisy image data b^δ and let $\widetilde{b}$ be the matrix reshape of the given image b^δ;
Choose a maximum window size $w_{\max}$;
for *each pixel location* (i,j) **do**
  $w = 3$;
  while $w \le w_{\max}$ **do**
    Compute $\widetilde{b}_{i,j}^{w,\min}$, $\widetilde{b}_{i,j}^{w,\max}$, and $\widetilde{b}_{i,j}^{w,\text{median}}$;
    if $\widetilde{b}_{i,j}^{w,\min} < \widetilde{b}_{i,j}^{w,\text{median}} < \widetilde{b}_{i,j}^{w,\max}$ **then**
      if $\widetilde{b}_{i,j}^{w,\min} < \widetilde{b}_{i,j} < \widetilde{b}_{i,j}^{w,\max}$ **then**
        The pixel $\widetilde{b}_{i,j}$ is not noisy;
        Exit **while** cycle;
      else
        $\widetilde{b}_{i,j} = \widetilde{b}_{i,j}^{w,\text{median}}$;
        Exit **while** cycle;
      end
    else
      $w = w + 2$;
      if $w > w_{\max}$ **then**
        $\widetilde{b}_{i,j} = \widetilde{b}_{i,j}^{w,\text{median}}$;
      end
    end
  end
end
Convert the matrix $\widetilde{b}$ into a stacked column vector $\widehat{b}^\delta$;

2) Directional Weighted Median Filter: This section summarizes properties of the directional weighted median filter (DWMF) proposed in [9]. Let $\widetilde{b}$ be defined as in the previous section, but assume now that it is corrupted by random-valued impulse noise. Then

$$\widetilde{b}_{i,j} = \begin{cases} \widehat{a}_{ij} & \text{with probability } \sigma, \\ b_{i,j} & \text{with probability } 1-\sigma, \end{cases}$$

where σ often is referred to as noise level and $\widehat{a}_{ij}$ is a random number chosen with uniform distribution in the interval $[\widetilde{b}_{\min}, \widetilde{b}_{\max}]$. The DWMF method detects noisy pixels by using the sum of weighted absolute differences between each pixel and its neighbors aligned in four main directions; see Figure 1. Then the noisy pixel is replaced using a weighted median operator constructed as

$$m = \text{median}\left\{\tilde{w} \diamond \widetilde{b}_\ell\right\},$$

where $\diamond$ is the repetition operator defined by $k \diamond a = \overbrace{a, a, \ldots\ldots, a}^{k \text{ times}}$ applied element-wise; the vector $\widetilde{b}_\ell \in \mathbb{R}^\ell$ contains the pixels of the adjacent neighbors in the four main directions, and $\tilde{w} \in \mathbb{N}^\ell$ assigns more weight to the pixels that are in the most clustered main direction. In detail, assume that we would like to denoise the $(0,0)$ pixel of an image y that

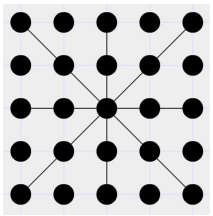


Fig. 1. The Four Main Directions of the DWMF.

is indexed on $\mathbb{Z}^2$. Define

$$\begin{aligned} S_1 &:= \{(-2,-2),(-1,-1),(0,0),(1,1),(2,2)\}, \\ S_2 &:= \{(0,-2),(0,-1),(0,0),(0,1),(0,2)\}, \\ S_3 &:= \{(2,-2),(1,-1),(0,0),(-1,1),(-2,2)\}, \\ S_4 &:= \{(-2,0),(-1,0),(0,0),(1,0),(2,0)\}. \end{aligned}$$

Each S_k denotes one of the four directions in Figure 1. Let $S_k^0 := S_k \setminus (0,0)$ and define $\Omega := \{(d_1,d_2) : -1 \le d_1, d_2 \le 1\}$. Compute the quantities, for $k = 1,\ldots,4$,

$$g^{(k)} = \sum_{(d_1,d_2)\in S_k^0} w_{d_1,d_2} |y_{d_1,d_2} - y_{0,0}|,$$

where

$$w_{d_1,d_2} = \begin{cases} 2, & (d_1,d_2) \in \Omega, \\ 1, & \text{otherwise.} \end{cases}$$

Let $\gamma = \arg\min_k\{g^{(k)}\}$. If $\gamma < T$, where T is a user-defined threshold, then we assume that the $(0,0)$ pixel is not affected by noise and we terminate the procedure; otherwise we apply the median filter as follows: For each direction, compute the standard deviations $\sigma^{(k)}$, i.e.,

$$(\sigma^{(k)})^2 = \frac{1}{4}\left[\sum_{(d_1,d_2)\in S_k^0} y_{d_1,d_2}^2 - \left(\sum_{(d_1,d_2)\in S_k^0} y_{d_1,d_2}\right)^2\right].$$

We determine which direction has the smallest standard deviation and denote it by C, i.e.,

$$C = \arg\min_k\{\sigma^{(k)}\},$$

and substitute the noisy pixel y by the weighted median given by

$$\hat{y} = \text{median}\left\{\tilde{w}_{d_1,d_2} \diamond y_{d_1,d_2} : (d_1,d_2) \in \Omega\right\},$$

where

$$\tilde{w}_{d_1,d_2} = \begin{cases} 2, & (d_1,d_2) \in S_C^0, \\ 1, & \text{otherwise.} \end{cases}$$

We repeat this procedure for each pixel in the image.

Algorithm 5 summarizes the computations for the directional weighted median filter. We set $T = 510$ in Algorithm 5; see below.

To improve the results, Algorithm 5 can be repeated with decreasing thresholds $T_z = 0.8T_{z-1}$, with $T_1 = 520$ and $z = 2,3,\ldots,N_{\max}$. These threshold choices are suitable for 8-bit gray-level images.

Algorithm 5: The DWMF method.

Consider the blurred and noisy image data b^δ and let $\widetilde{b}$ be the matrix reshape of the given image b^δ. Let T be a user-defined parameter;
Compute and store S_k^0 for $k = 1$ to 4;
for *each pixel location* (i, j) **do**
  for $k = 1$ *to* 4 **do**
    $g_{i,j}^{(k)} = \sum_{(d_1,d_2)\in S_k^0} w_{d_1,d_2} \left|\widetilde{b}_{i+d_1,j+d_2} - \widetilde{b}_{i,j}\right|$,
    where $w_{d_1,d_2} = \begin{cases} 2, & (d_1, d_2) \in \Omega \\ 1, & \text{otherwise} \end{cases}$;
  end
  $\gamma_{i,j} = \min_k \left\{ g_{i,j}^{(k)} \right\}$ for $k = 1$ to 4;
  if $\gamma_{i,j} > T$ **then**
    Calculate the standard deviation $\sigma_{i,j}^{(k)}$ of $\widetilde{b}_{i+d_1,j+d_2}$ with $(d_1, d_2) \in S_k^0$ and $k = 1$ to 4;
    $C_{i,j} = \arg\min_k \left\{ \sigma_{i,j}^{(k)} : k = 1 \text{ to } 4 \right\}$;
    $\widetilde{b}_{i,j} =$
    $\text{median}\left\{ \tilde{w}_{d_1,d_2} \diamond \widetilde{b}_{i+d_1,j+d_2} : (d_1, d_2) \in \Omega \right\}$
    where $\tilde{w}_{d_1,d_2} = \begin{cases} 2, & (d_1, d_2) \in S_{C_{i,j}}^0 \\ 1, & \text{otherwise} \end{cases}$;
  end
end
Convert the matrix $\widetilde{b}$ into a stacked column vector $\widehat{b^\delta}$;

D. MCV and GCV using Median Filters

We now describe our algorithms. To improve the quality of the images restored by the MM-GKS-MCV and MM-GKS-GCV methods, we preprocess the given images with median filters described in the previous sections. This gives the filtered data $\widehat{b^\delta}$. Then an edge-preserving restoration by MCV or GCV deblurs and further denoises the filtered data $\widehat{b^\delta}$. We determine $\widehat{b^\delta}$ from b^δ using Algorithm 4 when the noise is of salt-and-pepper-type, and we compute $\widehat{b^\delta}$ using Algorithm 5 when the noise in b^δ is random-valued impulse noise. Then $\widehat{b^\delta}$ is used in the minimization problem (2). We will refer to the MCV and GCV methods with preprocessing with a median filter as the M-MCV and M-GCV methods, respectively.

Note that since a median filter may fail to remove all the impulse or salt-and-pepper noise and does not remove the Gaussian noise, one has to regularize the minimization problem. Moreover, since some of the impulsive noise may still be present, it is beneficial to select $0 < p < 1$ in the ℓ^p-ℓ^q minimization problem.

III. Experiments and Comparisons

In this section, we present numerical examples to illustrate the performance of the proposed methods for image restoration. We compare the performance of the M-MCV and M-GCV methods to the performance of the MCV and GCV methods by showing images that have been restored by all these methods, as well as by computing the peak signal-to-noise ratio (PSNR) of the restored image. The PSNR for an image x is defined by

$$\text{PSNR}(x) = 10 \log_{10} \frac{255^2}{n\|x - x_{\text{true}}\|_2^2},$$

where x_{true} denotes the original image and 255 is the maximum value achievable by the entries of x_{true}. All images are represented by 8-bit pixels. Therefore, the pixel values are in the interval $[0, 255]$. To model salt-and-pepper noise, a certain percentage (noise level) of randomly chosen entries of the blurred but noise-free image b are set to 0 or 255. Impulse noise is modeled by letting a certain percentage (noise level) of randomly chosen entries of the blurred but noise-free image b be randomly chosen uniformly distributed integers in the interval $[0, 255]$. White Gaussian noise is modeled by pseudo-random numbers with a Gaussian distribution with zero mean and a specified variance. The noise level for this case is the ratio $\frac{\|\delta\|_2}{\|b\|_2}$, where δ is the error in b^δ. In the Peppers example, we consider salt-and-pepper noise of different percentages. Subsequently, we consider a mixture of random-valued impulse noise and white Gaussian noise for the Cameraman example, as well as a mixture of salt-and-pepper noise and white Gaussian noise for the Clock example.

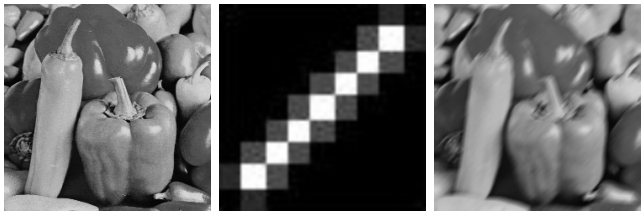
In Algorithm 4 and 5, one has to pad $\widetilde{b}$ before the first element and after the last element of each dimension by introducing new pixels around the edges of the image. This border provides space for handling of the boundary. For both the AMF and DWMF methods, we implement symmetric boundary padding. The maximum window size for the AMF is set to 39 throughout the tests. For the DWMF it suffices to set $N_{\max} = 6$, since repeating the DWMF on an image too many times may produce blurred edges.

We set $p = 0.8$ in all the examples, and we set $q = 0.1$ for the Peppers and Cameraman examples. This choice of q-value is explained in the Clock example, where restorations for different values of q are compared. Also, we fix $\varepsilon = 1$. This value is small compared to the elements of b^δ that are in the range $[0, 255]$. In Algorithm 2, we let $K = l = 10$, $d = \left[\frac{n}{200}\right]$, and let the μ_j be equispaced in a logarithmic scale. In Algorithms 1 and 3, we terminate the iterations either after 100 iterations or when two consecutive iterates are sufficiently close, i.e., when

$$\frac{\left\|x^{(k+1)} - x^{(k)}\right\|_2}{\left\|x^{(k)}\right\|_2} < 10^{-4}.$$

This choice of stopping criterion is suitable for many of the images that we considered.

The matrix A is the blurring operator. We represent the computed solution in terms of a framelet basis that is determined by linear B-splines similarly as in [3], and let the regularization operator L be the analysis opertor. The matrix L so defined is very sparse. This makes the matrix-vector product evaluations with L and L^T cheap; see [3], [5] for more information about the definition of L. The numerical examples are carried out in MATLAB R2020a running on a Windows 10 laptop computer with an i5-4210H CPU @ 2.40 GHz and 6GB of RAM.



(a) (b) (c)

Fig. 2. Peppers test case: (a) Original Peppers image (246×246 pixels) (b) PSF (9×9 pixels) (c) blurred image and noise-free image.

a) Pepper: In this example we consider the Peppers image of size 246×246 pixels shown in Figure 2(a) and we blur it with a PSF that simulates motion blur; see Figure 2(b). We obtain the blurred and noise-free image in Figure 2(c). We crop the boundaries to simulate realistic data; see [14] for more details on boundary conditions. We enforce reflexive boundary conditions. Salt-and-pepper noise is added at the levels of $20\%, 40\%, 55\%$, and 70%; see the first row of Figure 3.

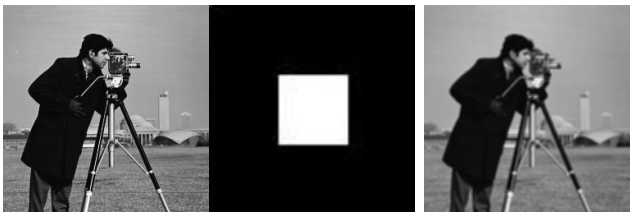
Table I shows the PSNR for each method and Figure 3 shows the restored images. We notice that M-GCV and M-MCV produce better restorations with higher PSNR compared to MCV and GCV. MCV cannot handle images with salt-and-pepper noise level higher than 20%. GCV produces oversmoothed reconstructions as the noise level increases. Both the M-MCV and M-GCV methods determine high-quality images. Overall, we observe that M-GCV yields a very accurate reconstruction which, in particular, is more accurate than the one obtained with M-MCV. These observations are confirmed by visual inspection of the reconstructions displayed in Figure 3 and by the PSNR values in Table I. Another advantage of M-GCV is that the GCV method demands much less CPU time than the MCV method. This is due to the fact that MCV requires several runs of the MM-GKS algorithm, while the GCV method only carries out a single run; see [3] for a discussion.

b) Cameraman: Our second example considers a 252×252 pixels image shown in Figure 4(a). The image is blurred by the average blur PSF displayed in Figure 4(b). We show the blurred and noise-free image in Figure 4(c). We added random-valued impulse noise with noise levels $20\%, 30\%, 40\%$, and 50%, as well as 1% white Gaussian noise. We enforce reflexive boundary conditions.

Figure 5 shows the blurred and noisy images (first row) and the restorations produced by each method. We report the PSNR-values in Table I. The approximate solutions computed with both GCV and MCV are not satisfactory. Moreover, for high random-valued impulse noise levels, the images reconstructed by M-MCV were not of high quality. M-GCV provides the best reconstructions in terms of suppressing noise and preserving edges and details. Again, M-GCV achieves a significantly higher PSNR than the other methods; see Table I.

c) Clock: For our final example, we use the clock image and blur it with an average blur; see Figure 6. Our goal here is to show the influence of q in the quality of the

Fig. 3. Blurred and noisy image and restorations for the Peppers test case. The first row reports the noisy and blurred image, the second row reports the restorations obtained with MM-GKS-MCV, the third row reports the restorations obtained with MM-GKS-GCV, the fourth row reports the restorations obtained with M-MCV, and the fifth row reports the restorations obtained with M-GCV. The noise increases from left to right, we considered the noise levels $20\%, 40\%, 55\%$, and 70%.



(a) (b) (c)

Fig. 4. Cameraman test case: (a) original image (252×252 pixels) (b) PSF (9×9 pixels) (c) blurred image and noise-free image.

TABLE I
PSNR OF RESTORED IMAGES OBTAINED FOR EACH NOISE LEVEL AND RESTORATION METHOD

Image	Noise Level	MCV	GCV	M-MCV	M-GCV
Peppers	20%	23.32	26.52	27.6	27.96
	40%	18.77	25.18	26.17	27.47
	55%	17.07	24.17	25.41	26.22
	70%	15.57	23.89	24.55	24.28
Cameraman	20%	22.89	22.33	25.64	28.09
	30%	19.53	21.24	22.92	26.22
	40%	18.13	23.89	22.70	25.05
	50%	16.75	21.55	23.21	23.86

Fig. 5. Blurred and noisy image and restorations cameraman test case. All the images are corrupted by a mixture of Gaussian and impulse noise. The first row reports the noisy and blurred image, the second row reports the restorations obtained with MM-GKS-MCV, the third row reports the restorations obtained with MM-GKS-GCV, the fourth row reports the restorations obtained with M-MCV, and the fifth row reports the restorations obtained with M-GCV. The noise increases from left to right, the Gaussian noise is always 1%, while we considered impulse noise levels $20\%, 30\%, 40\%$, and 50%.

TABLE II
PSNR OF THE RESTORED CLOCK IMAGE WITH DIFFERENT VALUES OF q OBTAINED USING M-GCV.

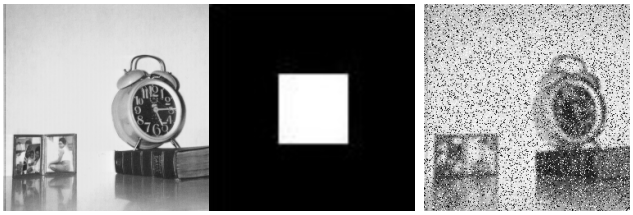
q	1	0.8	0.5	0.3	0.2	0.1
PSNR	27.38	27.85	26.17	27.11	28.39	28.60

reconstructed solutions in M-GCV. Therefore, we fix the noise. The image is corrupted by 20% salt-and-pepper noise and 3% white Gaussian noise. Similarly, we impose reflexive boundary conditions.

We vary the value of q to show how M-GCV performs for different values. Specifically, we let $q \in \{1, 0.7, 0.5, 0.3, 0.2, 0.1\}$. The restorations are shown in Figure 7. From visual inspection of the reconstructions, we notice that the noise gets less evident as q gets closer to zero. The PSNR in Table 2 confirms that M-GCV provides a better restoration for $q = 0.1$ than for larger values of q.

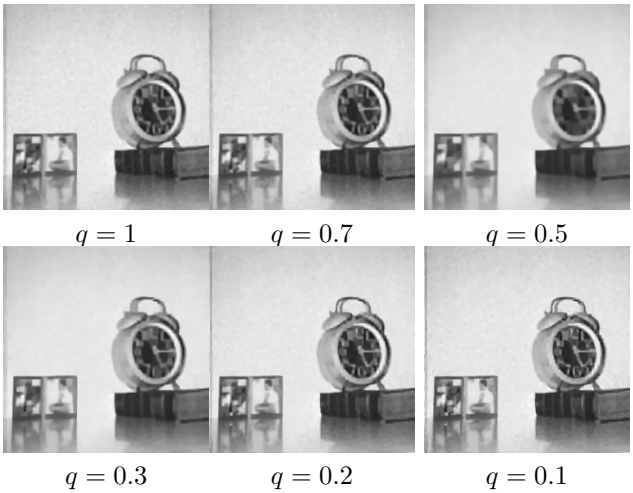
IV. CONCLUSIONS

We presented a two-phase method for restoring images that are contaminated by blur and impulse noise combined with Gaussian noise. The proposed approach is completely automatic and does not require any estimate of the regularization parameter. We showed that combining the majorization-minimization optimization framework with median filters and a modified version of cross validation and generalized cross validation produces high quality restorations. Numerical results show that M-GCV can handle noisy images contaminated by blur, Gaussian noise, and with up to 70% of the pixels contaminated by salt-and-pepper noise.



(a) (b) (c)

Fig. 6. Clock test case: (a) original image (252×252 pixels) (b) PSF (9×9 pixels) (c) blurred and noisy image (20% of salt-and-pepper noise and 3% white Gaussian noise).



$q = 1$ $q = 0.7$ $q = 0.5$

$q = 0.3$ $q = 0.2$ $q = 0.1$

Fig. 7. Restoration of the clock image with different values of q obtained using the M-GCV method. The image is corrupted by motion blur with 20% of salt-and-pepper noise and 3% white Gaussian noise.

REFERENCES

[1] A. Buccini, M. Pasha, and L. Reichel, "Modulus-based iterative methods for constrained ℓ_p-ℓ_q minimization," *Inverse Problems*, vol. 36, Art. 084001, 2020.

[2] A. Buccini, O. De la Cruz Cabrera, M. Donatelli, A. Martinelli, and L. Reichel, "Large-scale regression with non-convex loss and penalty," *Applied Numerical Mathematics*, vol. 157, pp. 590–601, 2020.

[3] A. Buccini and L. Reichel, "Generalized cross validation for ℓ^p-ℓ^q minimization," *Numerical Algorithms*, in press.

[4] ——, "An ℓ^2-ℓ^q regularization method for large discrete ill-posed problems," *Journal of Scientific Computing*, vol. 78, no. 3, pp. 1526–1549, 2019.

[5] ——, "An ℓ_p-ℓ_q minimization method with cross-validation for the restoration of impulse noise contaminated images," *Journal of Computational and Applied Mathematics*, vol. 375, Art. 112824, 2020.

[6] J.-F. Cai, R. H. Chan, and M. Nikolova, "Two-phase approach for deblurring images corrupted by impulse plus gaussian noise," *Inverse Problems & Imaging*, vol. 2, no. 2, pp. 187–204, 2008.

[7] R. H. Chan, C.-W. Ho, and M. Nikolova, "Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization," *IEEE Transactions on Image Processing*, vol. 14, no. 10, pp. 1479–1485, 2005.

[8] R. H. Chan and H.-X. Liang, "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*, A. Bruhn, T. Pock, and X.-C. Tai, Eds. Berlin: Springer, 2014, pp. 78–103.

[9] Y. Dong and S. Xu, "A new directional weighted median filter for removal of random-valued impulse noise," *IEEE Signal Processing Letters*, vol. 14, no. 3, pp. 193–196, 2007.

[10] C. Estatico, S. Gratton, F. Lenti, and D. Titley-Peloquin, "A conjugate gradient like method for p-norm minimization in functional spaces," *Numerische Mathematik*, vol. 137, no. 4, pp. 895–922, 2017.

[11] C. Fenu, L. Reichel, and G. Rodriguez, "GCV for Tikhonov regularization via global Golub–Kahan decomposition," *Numerical Linear Algebra with Applications*, vol. 23, no. 3, pp. 467–484, 2016.

[12] C. Fenu, L. Reichel, G. Rodriguez, and H. Sadok, "GCV for Tikhonov regularization by partial SVD," *BIT Numerical Mathematics*, vol. 57, no. 4, pp. 1019–1039, 2017.

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

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

[15] G. Huang, A. Lanza, S. Morigi, L. Reichel, and F. Sgallari, "Majorization–minimization generalized Krylov subspace methods for ℓ_p-ℓ_q optimization applied to image restoration," *BIT Numerical Mathematics*, vol. 57, no. 2, pp. 351–378, 2017.

[16] H. Hwang and R. A. Haddad, "Adaptive median filters: new algorithms and results," *IEEE Transactions on Image Processing*, vol. 4, no. 4, pp. 499–502, 1995.

[17] S. Kinderman, "Convergence analysis of minimization-based noise level-free parameter choice rules for linear ill-posed problems," *Electronic Transactions on Numerical Analysis*, vol. 38, pp. 233–257, 2011.

[18] J. Lampe, L. Reichel, and H. Voss, "Large-scale tikhonov regularization via reduction by orthogonal projection," *Linear Algebra and Its Applications*, vol. 436, pp. 2845–2865, 2012.

[19] A. Lanza, S. Morigi, L. Reichel, and F. Sgallari, "A generalized Krylov subspace method for ℓ_p-ℓ_q minimization," *SIAM Journal on Scientific Computing*, vol. 37, no. 5, pp. S30–S50, 2015.

[20] F. Sciacchitano, Y. Dong, and M. S. Andersen, "Total variation based parameter-free model for impulse noise removal," *Numerical Mathematics: Theory, Methods and Applications*, vol. 10, no. 1, pp. 186–204, 2017.