Inverse Problems and Imaging doi:10.3934/ipi.2021034

AN EFFICIENT MULTI-GRID METHOD FOR TV MINIMIZATION PROBLEMS

Zhenwei Zhang, Xue Li and Yuping Duan*
Center for Applied Mathematics
Tianjin University, Tianjin, China

Ke Yin*
Center for Mathematical Sciences
Huazhong University of Technology, Wuhan, China

Xue-Cheng Tai
Department of Mathematics
Hong Kong Bapist University, Kowloon Tong, Hong Kong, China

(Communicated by Tieyong Zeng)

Abstract. We propose an efficient multi-grid domain decomposition method for solving the total variation (TV) minimization problems. Our multi-grid scheme is developed based on the piecewise constant function spanned subspace correction rather than the piecewise linear one in [16], which ensures the calculation of the TV term only occurs on the boundaries of the support sets. Besides, the domain decomposition method is implemented on each layer to enable parallel computation. Comprehensive comparison results are presented to demonstrate the improvement in CPU time and image quality of the proposed method on medium and large-scale image denoising and reconstruction problems.

1. **Introduction.** Rudin, Osher and, Fatemi [28] first introduced the total variation into image processing for dealing with denoising problems. The Rudin-Osher-Fatemi model can restore a clean image from a noisy image f on a domain Ω in $\mathbb{R}^2$ by pursuing the minimizer of the following minimization problem

$$\min_{u\in \mathrm{BV}(\Omega)} F(u) := \left\{\frac{1}{2}\|u-f\|_2^2 + \alpha \mathrm{TV}(u)\right\}, \tag{1}$$

where α is a positive regularization parameter, $\mathrm{BV}(\Omega)$ is the space of functions of bounded variation, and the total variation of u is defined as

$$\mathrm{TV}(u) := \sup_{\boldsymbol{p}\in \mathrm{Y}} \int_\Omega u \mathrm{div} \boldsymbol{p} dx$$

2020 *Mathematics Subject Classification.* Primary: 65K10, 68U10; Secondary: 68K10.

Key words and phrases. Multi-grid method, domain decomposition, total variation, subspace correction, image denoising, image reconstruction.

The work of the first three authors were supported by NSFC 11701418 and 12071345, Major Science and Technology Project of Tianjin 18ZXRHSY00160 and Recruitment Program of Global Young Expert. The work of Yin was supported by NSFC 11801200 and a startup grant from HUST. The work of the last author was supported by Hong Kong Baptist University startup grants RG(R)-RC/17-18/02-MATH and FRG2/17-18/033.

* Corresponding author: yuping.duan@tju.edu.cn, kyin@hust.edu.cn.

with

$$\mathrm{Y} := \Big\{\boldsymbol{p} = (p^1, p^2) \in (C_0^1(\Omega))^2, \|\boldsymbol{p}\|_\infty \leq 1\Big\}.$$

Due to its abilities in removing noises as well as preserving sharp edges, TV has been well studied for various image processing applications including image reconstruction [33, 32], segmentation [12], inpainting [31], etc. Various methods have been developed for solving the Rudin-Osher-Fatemi model, which can be roughly divided into three categories according to the ways in constructing the minimization problem

(i) Primal methods: It is straightforward to find the minimizer of (1) by computing its Euler-Lagrange equation, which gives

$$-\alpha\nabla\cdot\Big(\frac{\nabla u}{\sqrt{|\nabla u|^2+\beta}}\Big) + (u-f) = 0, \tag{2}$$

with β being a small positive parameter to avoid the singularity. Different numerical methods have been used to solve the above nonlinear equation such as gradient descent method [28, 26], lagged fixed point iteration [1, 36, 37], explicit Euler method [26], etc. However, the aforementioned methods are slow in convergence due to strict constraints on the time step size. Other numerical solvers include graph cuts method [19], additive operator splitting scheme [27], etc.

(ii) Dual methods: Chambolle [5] solved the dual formulation of Rudin-Osher-Fatemi model

$$\inf_{\boldsymbol{p}\in\mathrm{Y}} \int_\Omega (\alpha \mathrm{div}\boldsymbol{p} - f)^2 dx, \tag{3}$$

where the semi-implicit gradient descent algorithm is used to solve the dual variable. The dual algorithm has been further studied for color TV minimization problems [3] and segmentation problem [21]. The dual algorithm for solving (3) is described as Algorithm 1.

Algorithm 1: Dual algorithm for solving the Rudin-Osher-Fatemi model

```
Input: Choose $\boldsymbol{p}^0$ and $\tau > 0$
Output: $u = f - \alpha \mathrm{div}\boldsymbol{p}$
for $n = 1, 2, \cdots,$ /* compute $\boldsymbol{p}$ */
 do
    (4)   $\boldsymbol{p}^{n+1} = \dfrac{\boldsymbol{p}^n + \tau\nabla(\mathrm{div}\boldsymbol{p}^n - f/\alpha)}{1 + \tau|\nabla(\mathrm{div}\boldsymbol{p}^n - f/\alpha)|};$
    if $\|\boldsymbol{p}^{n+1} - \boldsymbol{p}^n\|_2 < \epsilon\|\boldsymbol{p}^{n+1}\|_2$ then
        break;
    end
end
```

(iii) Primal-dual methods: The primal-dual approaches aim to solve both the primal and dual variables from a saddle-point problem, where the nonsmooth minimization problem (1) is decomposed into simpler subproblems. Chambolle and Pock [6] formulated a saddle-point problem based on Legendre-Fenchel conjugate, which reads

$$\min_{u\in\mathrm{BV}(\Omega)} \max_{\boldsymbol{p}} -\langle u, \mathrm{div}\boldsymbol{p}\rangle + \frac{1}{2}\|u-f\|_2^2 - \delta_{\|\cdot\|_\infty\leq\alpha}(\boldsymbol{p}), \tag{5}$$

with

$$\delta_{\|\cdot\|_\infty \le \alpha}(\boldsymbol{p}) = \begin{cases} 0, & \text{if } \max_{i,j} |\boldsymbol{p}_{i,j}| \le \alpha; \\ +\infty, & \text{else.} \end{cases} \tag{6}$$

The primal-dual algorithm for solving the Rudin-Osher-Fatemi model (1) is summarized in Algorithm 2. Other ways to formulate the saddle-point problem include the augmented Lagrangian type techniques [39], Douglas-Rachford splitting [18], etc. We refer readers to the review work [7] and references therein for more primal-dual algorithms for image processing problems. There are other variable splitting techniques, fast algorithms have been developed based on the Bregman distance [20] and penalty method [38].

Algorithm 2: Primal-dual method for solving the Rudin-Osher-Fatemi model

```
Input: Choose $\tau$, $\sigma > 0$, $\theta \in [0, 1]$
Output: $u$
for $n = 1, 2, \cdots,$ /* update $u^{n+1}$, $\boldsymbol{p}^{n+1}$, and $\bar{u}^{n+1}$ as follows */
do
    (7) $\begin{cases} \boldsymbol{p}^{n+1} = \frac{\boldsymbol{p}^n + \sigma\nabla\bar{u}^n}{\max(1, |\boldsymbol{p}^n + \sigma\nabla\bar{u}^n|/\alpha)}; \\ u^{n+1} = (u^n + \tau \mathrm{div}\boldsymbol{p}^{n+1} + \tau f)/(1+\tau); \\ \bar{u}^{n+1} = u^{n+1} + \theta(u^{n+1} - u^n); \end{cases}$
    if $\|u^{n+1} - u^n\|_2 < \epsilon\|u^{n+1}\|_2$ and $\|\boldsymbol{p}^{n+1} - \boldsymbol{p}^n\|_2 < \epsilon\|\boldsymbol{p}^{n+1}\|_2$ then
        break;
    end
end
```

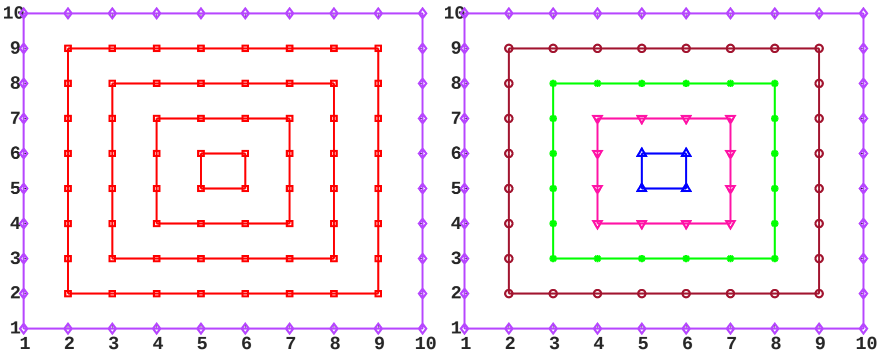
Although the aforementioned methods work well on small-scale and medium-scale problems, they usually suffer from high computational cost on large-scale problems. The multi-grid methods (MM) [40, 35] is a powerful numerical method for solving large-scale linear and nonlinear optimization problems, which have been successfully applied to image problems. The MM provides a hierarchical relaxation method to develop efficient algorithms for solving the linear and nonlinear elliptic equations. Some attempts have been made to develop MM for solving the curvature equation (2) derived from Rudin-Osher-Fatemi model [2, 29, 15]. Savage and Chen [30] presented a nonlinear multi-grid method based on the full approximation scheme for solving the nonlinear PDE (2). Chan and Chen [11] proposed a fast multilevel method using primal relaxations for the total variation image denoising and analyzed its convergence. Chan and Chen [10] proposed a multilevel method for the Rudin-Osher-Fatemi model based on piecewise constant refinements. Chen and Tai [16] developed an efficient nonlinear multi-grid method for solving the TV minimization model (1) using the piecewise linear function spanned subspace correction method. Chan and Chen [9] presented a fast multi-level algorithm for simultaneously denoising and deblurring image under the total variation regularization. Besides, Brito-Loeza and Chen [4] developed an efficient nonlinear multi-grid algorithm for solving the mean curvature based energy minimization model.

The domain decomposition method (DDM) is another promising technique to deal with large-scale problems, which divides the large problem into smaller problems for parallel computation. The DDM has also been successfully applied to the minimization problems modeling image denoising [22, 13, 24, 23], deblurring [41, 25] and segmentation [34, 17]. Hintermüller and Langer [22] developed a convergent nonoverlapping domain decomposition for the discrete dual problem of Rudin-Osher-Fatemi model. Chang *et al.* [13] proposed a convergent overlapping domain decomposition method for the dual problem (3) and estimated the rate of the convergence. Lee and Nam [24] analytically proved the convergence of the nonoverlapping domain decomposition methods to the minimizer of the global problem, which was later extended to the overlapping domain decomposition method in the continuous setting in [23]. Chen et al [14] introduced a highly parallel algorithm, which formulated the Rudin-Osher-Fatemi model as multiple overlapping, but independent, optimization problems. To the best of our knowledge, the study on the multi-grid domain decomposition method is still limited. In [42], the authors introduced the coarse mesh correction into the overlapping domain decomposition method for the Rudin-Osher-Fatemi model, where the smoother was required to overcome the non-smoothness of the total variation.

In this paper, we propose an efficient multi-grid domain decomposition algorithm for the Rudin-Osher-Fatemi model as well as general TV minimization image processing models. The essential idea is to use the piecewise constant function spanned subspaces correction method to construct the multi-grid algorithm by directly working on the primal variable to pursue its corrections over support sets/patches. The complexity of the proposed MM is $\mathcal{O}(N)$ with N being the total pixels of images, which is much less than $\mathcal{O}(N\log N)$ of the nonlinear MM in [16]. We further improve the efficiency of the proposed algorithm by applying the four-color domain decomposition method on each layer such that subproblems in the same color can be solved in parallel. Numerous numerical comparison experiments demonstrate that our proposal can save much CPU time in dealing with medium and large-scale image processing problems, which defeats both dual Algorithm 1 and primal-dual Algorithm 2 under the same stopping criteria. To sum up, our contributions include the following three aspects

1) We propose a novel multi-grid method based on the piecewise constant function spanned subspace correction and apply it to solve the TV minimization problems;
2) We develop an efficient multi-grid domain decomposition algorithm by implementing the non-overlapping domain decomposition on each layer for parallel computation, in which the computation of the TV term only occurs on the boundaries of the patches;
3) We discuss the applications of our multi-grid domain decomposition algorithm for general TV minimization problems including image restoration (denoising and deblurring problem) and image reconstruction (CT and MRI reconstruction).

The rest of the paper is organized as follows. Section 2 dedicates to review the nonlinear multi-grid method based on piecewise linear spanned subspaces for solving the Rudin-Osher-Fatemi model. In Section 3, we detail our multi-grid algorithm and multi-grid domain decomposition algorithm for image denoising problem. The multi-grid domain decomposition algorithm is extended to general TV minimization problems in Section 4. Numerical experiments are provided in Section 5 by comparing with state-of-the-art algorithms for solving image denoising and debluring problems, and CT and MRI reconstructions. We conclude the paper in Section 6 with some remarks and future works.



(a) The piecewise constant basis function (b) The piecewise linear basis function

FIGURE 1. An illustration of the piecewise linear basic function and piecewise constant basic function ϕ_j^i on $\bar{\tau}_j^i$ of the layer $j = 4$. In (a), the $\Box$ defines the weight of 1 for the piecewise constant function and $\Diamond$ defines the weight of 0 for the outer boundary $\partial\bar{\tau}_j^i$. In (b), the weights of $\circ$, $*$, $\triangledown$ and $\triangle$ are 1/8, 3/8, 5/8, 7/8 respectively.

2. **Preliminaries.**

Without loss of generality, a general convex minimization problem over a reflexive Banach space V can be given as follow

$$\min_{u\in V} F(u). \tag{8}$$

Suppose the space V is decomposed into J subspaces, i.e.,

$$V = V_1 + V_2 + \cdots + V_J. \tag{9}$$

Thus, for any $u \in V$, there exists $u_j \in V_j$ such that $u = \sum_{j=1}^J u_j$. The original minimization problem (8) can be reformulated as a series of the subspace minimization problems over each V_j such as

$$\min_{\delta u_j \in V_j} F(u + \delta u_j), \quad \text{for } j = 1, \cdots, J, \tag{10}$$

where u denotes a current approximation and δu_j is the correction on V_j. The classical iterative algorithms for solving (10) include Jacobi-type-method and Gauss-Seidel-type method [40], which are described as follows

- **Jacobi-type-method**: Choose an initial guess $u^0 \in V$ and relaxation parameters $\gamma_j > 0$ such that $\sum_{j=1}^J \gamma_j \leq 1$. Assume that $u^k \in V$ has been obtained and find $\delta u_j \in V_j$ in parallel for $j = 1, \cdots, J$ such that

$$F(u^k + \delta u_j) \leq F(u^k + v_j),\ \forall v_j \in V_j. \tag{11}$$

  Then u^{k+1} is defined by $u^{k+1} = u^k + \sum_{j=1}^J \gamma_j \delta u_j$.
- **Gauss-Seidel-type-method**: Let $u^0 \in V$ be given, and assume that $u^k \in V$ has been obtained. Then u^{k+1} is defined by $u^{k+j/J} = u^{k+(j-1)/J} + \delta u_j$, $\delta u_j \in V_j,\ j = 1, 2, \cdots, J$, such that

$$F(u^{k+(j-1)/J} + \delta u_j) \leq F(u^{k+(j-1)/J} + v_j), \forall v_j \in V_j. \tag{12}$$

Chen and Tai [16] designed a multi-grid method for solving Rudin-Osher-Fatemi model by the subspace correction method using the piecewise linear function. More specially, for a given domain $\Omega \subset \mathbb{R}^2$, we generate a nested sequence of uniform Cartesian partition $\mathcal{T}$ of Ω by a successive refinement process. Assume $\mathcal{T} = \mathcal{T}_J$ for some $J > 1$, we refine each rectangular element into 4 rectangular sub-mesh to get the next layer of fine mesh $\mathcal{T}_j$ for $j < J$, such that $\mathcal{T}_j = \{\tau_j^i\}$ of mesh size 2^{j-1} and $\Omega = \cup_{i=1}^{n_j} \tau_j^i$. Then, a finer partition is obtained by connecting the midpoints of the edges of the coarser grid and this process is repeated until we reach $\mathcal{T}_1$, which are the pixels of the image. For images whose sizes are not the power of 2, we prorogate them before the above procedure using boundary conditions. It is straightforward to define V_j as a finite element space

$$V_j = \{v : \ v|_\tau \in \mathcal{P}_l(\tau), \ \forall \tau \in \mathcal{T}_j\}; \tag{13}$$

where $\mathcal{P}_l$ denotes the space of piecewise linear functions, and the finite element space V_j is equipped by a nodal basis $\{\phi_j^i\}_{i=1}^{n_j}$. An illustration of the basis function ϕ_j^i on the coarse layer $j = 4$ is shown in Figure 1 (a). More details of the basis functions in the piecewise linear function spaces can be found in [16]. Associated with each nodal basic function, the one dimensional subspace can be defined by $V_j^i = \text{span}\{\phi_j^i\}$. Then the subspace V_j can be described as the direct sum of all subspaces V_j^i, such that

$$V_j = \sum_{i=1}^{n_j} V_j^i.$$

The subspace correction refers to find $\delta u_j = \sum_{i=1}^{n_j} c_j^i \phi_j^i$ for $\delta u \in V_j$, where c_j^i is estimated by the fixed point method [16]. The corresponding whole space V is defined as

$$V = \sum_{j=1}^{J} \sum_{i=1}^{n_j} V_j^i.$$

If F is convex and continuous, u^k obtained by both Jacobi and Gauss-Seidel method is theoretically convergent to one minimizer of F. The convergence rate has also been estimated when F is Lipschitz continuous and strongly convex. More details of the convergence results can be found in [35].

3. The new multi-grid method.

In this section, we consider developing an efficient multi-grid method for solving the discrete Rudin-Osher-Fatemi model as follows

$$F(u) := \sum_{x \in \Omega} \frac{1}{2}(u(x) - f(x))^2 + \alpha \sum_{x \in \Omega} \sqrt{|\nabla u(x)|^2 + \beta}, \tag{14}$$

where $\nabla u := (D_x^+ u, D_y^+ u)$ with D_x^+ and D_y^+ being the standard forward finite difference operator, and $\beta > 0$ is introduced to avoid the singularity.

3.1. The multi-grid method using piecewise constant basis functions.

In our method, we define V_j as space of the piecewise constant functions such as

$$V_j = \{v : \ v|_\tau \in \mathcal{P}_c(\tau), \ \forall \tau \in \mathcal{T}_j\}, \tag{15}$$

where $\mathcal{P}_c$ denotes the space of all piecewise constant functions. We equip the piecewise constant function space V_j with a set of basis functions $\{\phi_j^i\}_{i=1}^{n_j}$, which is defined as

$$\phi_j^i(x) = \begin{cases} 1, & \text{if } x \in \tau_j^i; \\ 0, & \text{if } x \notin \tau_j^i; \end{cases} \qquad i = 1, \cdots, n_j. \tag{16}$$

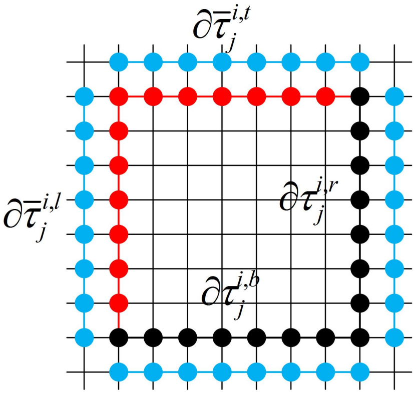


FIGURE 2. An illustration of $\partial\bar{\tau}_j^{i,l}$, $\partial\bar{\tau}_j^{i,t}$, $\partial\tau_j^{i,b}$ and $\partial\tau_j^{i,r}$ located on $\bar{\tau}_j^i$ of the layer $j = 4$.

The illustration of the basis function (16) is shown in Figure 1 (b).

Let Ω_j^i be the support set of ϕ_j^i and $\bar{\Omega}_j^i$ be its closure. We define $\tau_j^i = \{x \in \Omega_j^i \cap \Theta\}$ and $\bar{\tau}_j^i = \{x \in \bar{\Omega}_j^i \cap \Theta\}$ with Θ being the set of the nodal points on the finest mesh. Our target is to find the correction $c \in \mathbb{R}$ on subspaces V_j^i, for $i = 1, \ldots, n_j$, $j = 1, \ldots, J$. As shown in Figure 2, for piecewise constant basis functions ϕ_j^i on the coarse level $j = 4$, the computation of $\nabla\phi_j^i(x)$ only occurs on boundaries of τ_j^i. More specifically, $D_x^+\phi_j^i(x)$ is computed on the set $x \in \{\partial\bar{\tau}_j^{i,l}, \partial\tau_j^{i,r}\}$ and $D_y^+\phi_j^i(x)$ is computed on the set $x \in \{\partial\bar{\tau}_j^{i,t}, \partial\tau_j^{i,b}\}$, respectively. Then the energy problem (14) can be simplified into the following form

$$\begin{aligned}
F(u + c\phi_j^i) &= \sum_{x\in\Omega\setminus\tau_j^i} \frac{1}{2}(u(x) - f(x))^2 + \alpha \sum_{x\in\Omega\setminus\bar{\tau}_j^i} \sqrt{|\nabla u(x)|^2 + \beta} \\
&\quad + \sum_{x\in\tau_j^i} \frac{1}{2}(u(x) + c\phi_j^i(x) - f(x))^2 + \alpha \sum_{x\in\bar{\tau}_j^i} \sqrt{|\nabla(u(x) + c\phi_j^i(x))|^2 + \beta} \\
&= \widetilde{F_j^i}(u) + \frac{s}{2}(c - f^*)^2 + \alpha \sum_{x\in\{\partial\bar{\tau}_j^i, \partial\tau_j^i\}} \sqrt{(D_x^+ u + cD_x^+\phi_j^i)^2 + (D_y^+ u + cD_y^+\phi_j^i)^2 + \beta},
\end{aligned}$$

where $\widetilde{F_j^i}(u)$ contains all terms without c, and

$$f^* = \sum_{x\in\tau_j^i} \bar{f}(x)/s \quad \text{with} \quad \bar{f} = f - u,$$

and $s = \sum_{x\in\tau_j^i} \phi_j^i$ is the number of elements contained in τ_j^i. Therefore, minimizing the above energy functional is equivalent to solve

$$\min_{c\in\mathbb{R}} \frac{s}{2}(c - f^*)^2 + \alpha \sum_{x\in\{\partial\bar{\tau}_j^i, \partial\tau_j^i\}} \sqrt{(D_x^+ u + cD_x^+\phi_j^i)^2 + (D_y^+ u + cD_y^+\phi_j^i)^2 + \beta}, \tag{17}$$

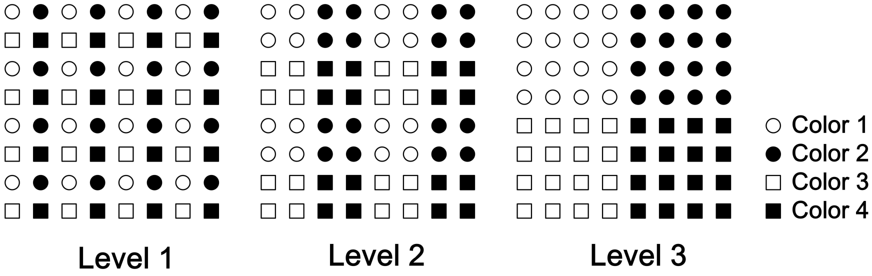


FIGURE 3. Illustration of the 4-color domain decomposition for a domain of size 8×8, where each color has one element on the coarse layer $j = 3$.

the minimizer of which can be obtained by the fixed point iteration as follows

$$\left(\alpha \sum_{x\in\{\partial\bar{\tau}_j^i,\partial\tau_j^i\}} \frac{|D_x^+\phi_j^i(x)|^2+|D_y^+\phi_j^i(x)|^2}{\sqrt{|D_x^+(u(x)+c^\ell\phi_j^i(x))|^2+|D_y^+(u(x)+c^\ell\phi_j^i(x))|^2+\beta}}+s\right)c^{\ell+1}$$
$$= sf^* - \alpha \sum_{x\in\{\partial\bar{\tau}_j^i,\partial\tau_j^i\}} \frac{D_x^+u(x)\cdot D_x^+\phi_j^i(x)+D_y^+u(x)\cdot D_y^+\phi_j^i(x)}{\sqrt{|D_x^+(u(x)+c^\ell\phi_j^i(x))|^2+|D_y^+(u(x)+c^\ell\phi_j^i(x))|^2+\beta}}, \tag{18}$$

for $\ell = 0, 1, 2, \ldots$. Therefore, we can use the Gauss-Seidel-type of multi-grid method to solve the Rudin-Osher-Fatemi model (14). The main difference between our MM and [16] is that the computation on each subspace only happens on the boundaries rather than the entirety of the support sets.

3.2. **Domain decomposition on the multi-grid scheme.** For implementation, we find that much time is consumed due to the sequential computation of V_j^i, $i = 1, \ldots, n_j$. Therefore, the non-overlapping domain decomposition method is applied on each layer to realize parallel computation for dealing with large-scale problems.

Figure 3 displays the four-color decomposition on each layer from $j = 1$ to $j = 3$. More specifically, we divide the basis function $\{\phi_j^i\}_{i=1}^{n_j}$ into four groups $\cup_{k=1}^4\{\phi_j^i : i \in I_k\}$ to reduce the dependency on the order of basis functions and improve the parallelism for subproblems on each layer, where I_k contains the indexes with the same color. This decomposition guarantees that neighboring patches in a 4-connected neighborhood are in different subsets. We can see that the support of the basic functions $\{\phi_j^i : i \in I_k\}$ is non-overlapping for each $k = 1, 2, 3, 4$, and the minimization of $F(u + c\phi_j^i)$ for $i \in I_k$ can be solved in parallel. In particular, four subproblems are solved in consecutive order

$$\min_{\delta u\in V_j^{(k)}} F(u+\delta u), \quad \text{for } k = 1, 2, 3, 4,$$

where $V_j^{(k)} = \operatorname{span}\{\phi_j^i : i \in I_k\}$ and $V_j = \sum_{k=1}^4 V_j^{(k)}$. It is readily checked that

$$\min_{\delta u\in V_j^{(k)}} F(u+\delta u) = \min_{\delta u_i\in V_j^{(k)}} F(u+\sum_{i\in I_k}\delta u_i) = \min_{\boldsymbol{c}\in\mathbb{R}^{N_k}} F(u+\sum_{i\in I_k} c_j^i\phi_j^i), \tag{19}$$

where $\boldsymbol{c} = (c_j^1, c_j^2, \cdots, c_j^{N_k})$ with N_k being the total number of elements in I_k.

Since the supports of the basis functions $\{\phi_j^i : i \in I_k\}$ are non-overlapping, the subproblems for all $i \in I_k$ can be solved by the fixed point method in parallel, where we are required to solve a system of equations as follows

$$\begin{bmatrix} d_{11} & 0 & \cdots & 0 & 0 \\ 0 & d_{22} & \cdots & 0 & 0 \\ & & \vdots & & \\ 0 & 0 & \cdots & d_{N_k-1,N_k-1} & 0 \\ 0 & 0 & \cdots & 0 & d_{N_k,N_k} \end{bmatrix} \begin{bmatrix} c_j^1 \\ c_j^2 \\ \vdots \\ c_j^{N_k-1} \\ c_j^{N_k} \end{bmatrix} = \begin{bmatrix} b_j^1 \\ b_j^2 \\ \vdots \\ b_j^{N_k-1} \\ b_j^{N_k} \end{bmatrix}, \tag{20}$$

where

$$d_{i,i} = s + \alpha \sum_{x \in \{\partial \bar{\tau}_j^i, \partial \tau_j^i\}} \frac{|D_x^+ \phi_j^i(x)|^2 + |D_y^+ \phi_j^i(x)|^2}{\sqrt{|D_x^+(u(x) + (c_j^i)^\ell \phi_j^i(x))|^2 + |D_y^+(u(x) + (c_j^i)^\ell \phi_j^i(x))|^2 + \beta}},$$

$$b_j^i = sf^* - \alpha \sum_{x \in \{\partial \bar{\tau}_j^i, \partial \tau_j^i\}} \frac{D_x^+ u(x) \cdot D_x^+ \phi_j^i(x) + D_y^+ u(x) \cdot D_y^+ \phi_j^i(x)}{\sqrt{|D_x^+(u(x) + (c_j^i)^\ell \phi_j^i(x))|^2 + |D_y^+(u(x) + (c_j^i)^\ell \phi_j^i(x))|^2 + \beta}},$$

for $i = 1, \cdots, N_k$. The parallel four-color subspace correction algorithm for solving one layer Rudin-Osher-Fatemi model (14) is summarized in Algorithm 3.

Algorithm 3: Parallel four-color subspace correction

```
/* solve $\min_{\{c_j^1,\dots,c_j^{n_j}\} \in \mathbb{R}^{n_j}} F(u + \sum_{i=1}^{n_j} c_j^i \phi_j^i)$ */
Input: $\{\phi_j^i\}_{i=1}^{n_j} = \cup_{k=1}^4 \{\phi_j^i : i \in I_k\}$, $u^0$
Output: $\{c_j^1, \dots, c_j^{n_j}\}$
for $t = 0$ to $N_c$, /* iterative process */
do
    for $k = 1$ to $4$, /* four-color iterations */
    do
        for $\ell = 0, 1, \dots, \ell_{max}$ do
            $\boldsymbol{c}^{\ell+1} = \arg\min_{\boldsymbol{c} \in \mathbb{R}^{N_k}} F(u^{t+(k-1)/4} + \sum_{i \in I_k} (c_j^i)^\ell \phi_j^i)$;
            if $|\boldsymbol{c}^{\ell+1} - \boldsymbol{c}^\ell|_\infty < \epsilon$ then
                break;
            end
        end
        $u^{t+k/4} \leftarrow u^{t+(k-1)/4} + \sum_{i \in I_k} c_j^i \phi_j^i$;
    end
    if $\|u^{t+1} - u^t\|_2 < \epsilon_{inner} \|u^{t+1}\|_2$ then
        break;
    end
end
```

3.3. **V-cycle based multi-grid method for solving Rudin-Osher-Fatemi model.** In the multi-grid approach, V-cycle is a common method. That is the solution to the minimization model is updated from coarsest layer V_J down to the finest layer V_1, then from the finest to the coarsest layer, which forms a complete V-cycle. In practice, we find that half of the V-cycle is sufficient for the decrease of the energy functional F, while the other half of the V-cycle does little improvement. Thus the fine to coarse subspace correction can be omitted.

Thus, the implementation of the algorithm to solve the minimization model (1) is described in Algorithm 4.

Algorithm 4: Multi-grid method for solving Rudin-Osher-Fatemi model (14)

```
/* solve $\min_{\{c_1^1,\ldots,c_1^{n_1},\ldots,c_J^1,\ldots c_J^{n_J}\}\in\mathbb{R}^{n_1\times\cdots\times n_J}} F(u+\sum_{j=1}^{J}\sum_{i=1}^{n_j} c_j^i\phi_j^i)$ */
Input: $f$, $u^0$, $\alpha$, $\epsilon$, $N_w$, $N_c$
Output: $u$.
for $t = 1$ to $N_w$ /* outer iterations */
do
    for $j = J$ to $1$ /* coarse to fine */
    do
        /* four-color subspace correction */
        $\{c_j^1,\ldots,c_j^{n_j}\} = \arg\min_{\{c_j^1,\ldots,c_j^{n_j}\}\in\mathbb{R}^{n_j}} F(u^{t+(J-j)/J}+\sum_{i=1}^{n_j} c_j^i\phi_j^i)$ by
         Algorithm 3;
        $u^{t+(J-j+1)/J} \leftarrow u^{t+(J-j)/J}+\sum_{i=1}^{n_j} c_j^i\phi_j^i$;
    end
    if $\|u^{t+1}-u^t\|_2 < \epsilon_{outer}\|u^{t+1}\|_2$ then
        break;
    end
end
```

3.4. **Convergence.** In this subsection, we show that the Rudin-Osher-Fatemi energy functional $F(u)$ is non-increasing in Algorithm 4.

Lemma 3.1. *Let u^t be the solution after t outer iteration in Algorithm 4, then $F(u^t) \le F(u^{t-1})$.*

Proof. By the construction of the solution in each iteration, for fixed level j, there is

$$u^t = u^{t-1} + \sum_{k=1}^{4}\sum_{i\in I_k} c_j^i\phi_j^i, \tag{21}$$

Suppose $\tilde{u}$ is the current solution when u is updated on $\tau_j^i = \mathrm{supp}(\phi_j^i)$. Therefore, we have

$$F(\tilde{u}+c_j^i\phi_j^i) \le F(\tilde{u}). \tag{22}$$

Because τ_j^i is non-overlapping for all $i \in I_k$, $k = 1,2,3,4$, we have $F(u^t) \le F(u^{t-1})$. □

3.5. **Computational complexity.** We briefly discuss the complexity of our MM as follows. Suppose the size of the image is $N = m \times n$ and rectangular grids are used on all coarse layers. The number of total patches on level j is $m_j \times n_j$ such that $m_j = m/2^{j-1}$ and $n_j = n/2^{j-1}$. Let ℓ_{max} denote the number of fixed point iteration for solving the correction c. Then the total cost for subspace problem is about

$32\ell_{max} \times 2^{j-1} \times m_j n_j = 32\ell_{max} N 2^{1-j}$ floating point operations (FLO), where the 2^{j-1} means the number of $\partial\bar{\tau}_j^i$. Computing $\bar{f}$ requires $N = mn$ FLO and the FLO for computing f^* is $2b_j n_j m_j \approx 2N$ since the size of the set $\bar{\tau}_j^i$ is $b_j = 2^{j-1} \times 2^{j-1}$. Then the number of FLO over all J levels is $\mathcal{O}(\sum_{j=1}^{J}(1+2+32\ell_{max}2^{1-j})N) = \mathcal{O}(N)$.

On the other hand, the cost of the fixed point iteration (18) in [16] is $32\ell_{max} \times b_j \times m_j n_j \approx 32\ell_{max} N$ FLO, where $b_j = 2^{j-1} \times 2^{j-1}$ is the size of the set I_j^i. Thus the number of FLO over all J levels $\mathcal{O}(\sum_{j=1}^{J}[1+2+32\ell_{max}]N) \approx \mathcal{O}(NJ) \approx \mathcal{O}(NlogN)$ FLO since $\max(J) \leq \log_2 \min(m, n) \leq \log N$. Thus, our MM is more efficient than the MM in [16], which will be further verified by the numerical experiments.

4. The multi-grid method for TV based image processing problems.

In this section, the multi-grid method is used to solve general image restoration and image reconstruction problems. The task is to recover $u \in \mathbb{R}^2$ from the observed data defined by

$$b = Ku + \nu, \tag{23}$$

where ν is the random noise and K is a linear and bounded operator varying with different image processing tasks. To be specific, K denotes a convolution operator for image deblurring, K represents Radon transform and Fourier transform for CT and MRI reconstruction, respectively. Then the TV regularization model for dealing with the above general inverse problem can be formulated as follows

$$F(u) := \frac{1}{2}\|Ku - b\|_2^2 + \alpha \sum_{x\in\Omega} \sqrt{|\nabla u(x)|^2 + \beta}. \tag{24}$$

Similarly, we implement the non-overlapping domain decomposition method on each layer to solve the subproblems of the same color simultaneously. For each color in V_j, the corresponding minimization problem over the subspace is given as

$$\min_{c_j \in \mathbb{R}^{N_k}} \frac{1}{2}\|K(u + \sum_{i\in I_k} c_j^i \phi_j^i) - b\|_2^2 + \alpha \sum_{i\in I_k} \sum_{x\in\bar{\tau}_j^i} \sqrt{|\nabla(u(x) + c_j^i \phi_j^i(x))|^2 + \beta}, \tag{25}$$

which can be also solved by applying the fixed point method to its Euler-Lagrange equation. The fixed point iteration is written as the following system of equations (26)

$$\begin{bmatrix} d_{1,1} & \langle K\phi_j^1, K\phi_j^2\rangle & \cdots & \langle K\phi_j^1, K\phi_j^{N_k}\rangle \\ \langle K\phi_j^2, K\phi_j^1\rangle & d_{2,2} & \cdots & \langle K\phi_j^2, K\phi_j^{N_k}\rangle \\ \vdots & \vdots & & \vdots \\ \langle K\phi_j^{N_k-1}, K\phi_j^1\rangle & \langle K\phi_j^{N_k-1}, K\phi_j^2\rangle & \cdots & \langle K\phi_j^{N_k-1}, K\phi_j^{N_k}\rangle \\ \langle K\phi_j^{N_k}, K\phi_j^1\rangle & \langle K\phi_j^{N_k}, K\phi_j^2\rangle & \cdots & d_{N_k,N_k} \end{bmatrix} \begin{bmatrix} c_j^1 \\ c_j^2 \\ \vdots \\ c_j^{N_k-1} \\ c_j^{N_k} \end{bmatrix} = \begin{bmatrix} b_j^1 \\ b_j^2 \\ \vdots \\ b_j^{N_k-1} \\ b_j^{N_k} \end{bmatrix},$$

where

$$d_{i,i} = \langle K\phi_j^i, K\phi_j^i\rangle + \alpha \sum_{x\in\{\partial\bar{\tau}_j^i, \partial\tau_j^i\}} \frac{\langle \nabla\phi_j^i, \nabla\phi_j^i\rangle}{\sqrt{|D_x^+(u(x) + (c_j^i)^\ell \phi_j^i(x))|^2 + |D_y^+(u(x) + (c_j^i)^\ell \phi_j^i(x))|^2 + \beta}},$$

$$b_j^i = \langle b - Ku, K\phi_j^i\rangle - \alpha \sum_{x\in\{\partial\bar{\tau}_j^i, \partial\tau_j^i\}} \frac{\langle \nabla\phi_j^i, \nabla u\rangle}{\sqrt{|D_x^+(u(x) + (c_j^i)^\ell \phi_j^i(x))|^2 + |D_y^+(u(x) + (c_j^i)^\ell \phi_j^i(x))|^2 + \beta}},$$

for $i = 1, \cdots, N_k$. For such a symmetric linear system, we use the conjugate gradient as a fast numerical solver.

256×256 256×256 512×512 512×512

1024×1024 1024×1024 2048×2048 2048×2048

FIGURE 4. The test images with sizes and their identifiers.

5. Numerical experiments.

In this section, we evaluate the performance of our proposed multi-grid algorithm on image denoising, CT reconstruction, and MRI reconstruction problems. All of the experiments were performed in MATLAB R2016a running on a desktop with an Intel Core i5 CPU at 3.3 GHz and 8 GB memory. The quality of the reconstruction results are measured by the Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) defined as

$$\begin{aligned} \mathrm{PSNR}(u,\underline{u}) &= 10\log_{10}\frac{\underline{u}_{\max}^2\cdot mn}{\|u-\underline{u}\|^2}\mathrm{dB},\\ \mathrm{SSIM}(u,\underline{u}) &= \frac{(2M_uM_{\underline{u}}+C_1)+(2\sigma_{u\underline{u}}+C_2)}{(M_u^2+M_{\underline{u}}^2+C_1)(\sigma_u^2+\sigma_{\underline{u}}^2+C_2)}, \end{aligned} \tag{27}$$

where u and $\underline{u}$ denote the restored and original images, respectively, $\underline{u}_{\max}$ represents the possible maximum pixel value of the image, M_u and $M_{\underline{u}}$ are the mean value of u and $\underline{u}$, σ_u and $\sigma_{\underline{u}}$ are variance, $\sigma_{u\underline{u}}$ is the covariance of u and $\underline{u}$, and C_1, C_2 are two constants.

5.1. Image denoising.

In this section, we evaluate our multi-grid Algorithm 4 (denoted by MMC) on image denoising problems by comparing it with the piecewise linear multi-grid method (denoted by MML) in [16], the dual Algorithm 1, and the primal-dual Algorithm 2. Note that the domain decomposition method has been applied to MML for a fair comparison. The implementation details of each algorithm are given as follows:

1) MMC/MML: The maximum level J is set to the same value for both MMC and MML, which is selected to achieve best convergence. We set $N_c = N_w = 100$, $\epsilon_{inner} = 10^{-2}$, $\epsilon_{outer} = 10^{-4}$. The iteration of the fixed point scheme is terminated either the number of iterations reaching 10 or $\|c^\ell - c^{\ell-1}\|_\infty < 0.1$. Besides, β is chosen as $\beta = 10^{-6}$ for all experiments.
2) Dual method: The step size of the dual variable $\boldsymbol{p}$ is set as $\tau = 1/4$. We use the relative error of dual variable as the stopping criterion, which is $\|\boldsymbol{p}^{t+1}-\boldsymbol{p}^t\|_2 < \epsilon\|\boldsymbol{p}^{t+1}\|_2$ with $\epsilon = 10^{-4}$.
3) Primal-dual method: The step sizes are set as $\tau = 1/4$ and $\sigma = 1/(8\tau)$ for the primal and dual variable, respectively. The iteration is terminated when both

Table 1. The results on selected images (#1, #3, #5 and #7) with different J for noise level $\sigma = 20$.

	α	10			15			20		
MML	J	Iter	Energy	CPU(s)	Iter	Energy	CPU(s)	Iter	Energy	CPU(s)
#1	1	54	2.32E7	**1.10**	86	3.12E7	**2.05**	128	3.80E7	**3.12**
	2	23	2.31E7	1.21	50	3.05E7	2.54	80	3.62E7	3.49
	3	24	**2.29E7**	1.48	40	**3.03E7**	2.46	31	3.53E7	3.67
	4	22	2.29E7	1.53	32	3.03E7	3.35	29	**3.52E7**	3.21
	5	24	2.29E7	1.85	32	3.03E7	4.88	29	3.52E7	4.51
	6	22	2.29E7	2.08	23	3.03E7	5.73	29	3.52E7	5.71
#3	1	35	8.60E7	**5.64**	82	1.14E8	11.32	139	1.38E8	19.35
	2	18	8.57E7	5.87	36	1.11E8	10.32	65	1.29E8	16.43
	3	18	**8.56E7**	6.18	44	**1.07E8**	10.65	32	1.25E8	14.47
	4	18	8.56E7	6.79	31	1.07E8	**9.31**	32	**1.24E8**	**11.28**
	5	18	8.56E7	6.77	31	1.07E8	9.78	32	1.24E8	12.02
	6	18	8.56E7	7.25	31	1.07E8	10.17	32	1.24E8	12.56
#5	1	49	3.54E8	**27.32**	71	4.72E8	**42.35**	198	5.63E8	79.56
	2	31	3.54E8	31.47	53	4.60E8	50.61	70	5.34E8	71.35
	3	28	**3.53E8**	32.06	43	**4.59E8**	49.35	54	5.24E8	60.21
	4	21	3.53E8	33.37	36	4.59E8	48.35	52	**5.22E8**	**52.13**
	5	21	3.53E8	34.25	36	4.59E8	50.22	52	5.22E8	66.32
	6	21	3.53E8	38.24	36	4.59E8	51.78	52	5.22E8	75.21
#7	1	57	1.38E9	121.32	120	1.82E9	237.21	189	2.21E9	369.75
	2	31	1.38E9	117.24	61	1.76E9	225.76	102	2.05E9	366.14
	3	26	**1.37E9**	**115.78**	44	1.76E9	193.35	65	1.99E9	299.78
	4	26	1.37E9	122.73	39	**1.75E9**	**192.49**	56	**1.94E9**	**280.32**
	5	26	1.37E9	126.25	39	1.75E9	208.15	53	1.94E9	288.25
	6	26	1.37E9	139.35	39	1.75E9	229.29	53	1.94E9	282.12
MMC	J	Iter	Energy	CPU(s)	Iter	Energy	CPU(s)	Iter	Energy	CPU(s)
#1	1	33	2.31E7	0.85	72	3.12E7	1.72	112	3.79E7	2.77
	2	21	**2.31E7**	**0.69**	41	3.05E7	1.22	37	3.63E7	1.21
	3	23	2.31E7	0.83	33	**3.03E7**	**1.15**	46	3.56E7	1.54
	4	21	2.31E7	0.88	33	3.03E7	1.21	34	**3.54E7**	**1.20**
	5	21	2.31E7	0.83	29	3.03E7	1.31	33	3.54E7	1.28
	6	20	2.31E7	0.82	23	3.03E7	1.95	33	3.54E7	1.55
#3	1	39	8.59E7	4.96	80	1.14E8	9.65	134	1.39E8	15.84
	2	28	**8.58E7**	**4.07**	36	**1.11E8**	**5.13**	54	1.31E8	7.54
	3	27	8.58E7	4.25	42	1.07E8	6.41	46	1.26E8	6.99
	4	27	8.58E7	4.27	37	1.07E8	5.97	34	**1.24E8**	**5.75**
	5	27	8.58E7	4.73	33	1.07E8	5.68	38	1.24E8	6.56
	6	26	8.58E7	4.56	30	1.07E8	5.93	32	1.24E8	6.73
#5	1	51	3.54E8	23.32	99	4.71E8	44.23	161	5.64E8	74.36
	2	33	3.54E8	**20.67**	49	4.60E8	29.95	66	5.37E8	42.74
	3	31	**3.53E8**	23.07	46	4.59E8	31.26	47	5.26E8	34.33
	4	30	3.53E8	22.07	40	**4.59E8**	**28.98**	48	**5.23E8**	**34.28**
	5	28	3.53E8	25.88	42	4.59E8	31.20	43	5.23E8	37.25
	6	29	3.53E8	30.15	39	4.59E8	36.59	44	5.23E8	42.67
#7	1	57	1.38E9	121.96	120	1.82E9	237.88	189	2.21E9	369.02
	2	36	1.38E9	100.06	56	1.76E9	124.64	84	2.05E9	183.47
	3	34	1.38E9	86.18	44	1.75E9	110.30	62	1.99E9	150.35
	4	34	**1.36E9**	**81.35**	43	**1.74E9**	**105.42**	43	**1.95E9**	**114.78**
	5	37	1.36E9	90.06	43	1.74E9	109.73	39	1.95E9	117.75
	6	36	1.36E9	98.81	43	1.74E9	115.06	40	1.95E9	123.42

of the following criteria are met: $\|u^{t+1} - u^t\|_2 < \epsilon\|u^{t+1}\|_2$, $\|\boldsymbol{p}^{t+1} - \boldsymbol{p}^t\|_2 < \epsilon\|\boldsymbol{p}^{t+1}\|_2$ with $\epsilon = 10^{-4}$.

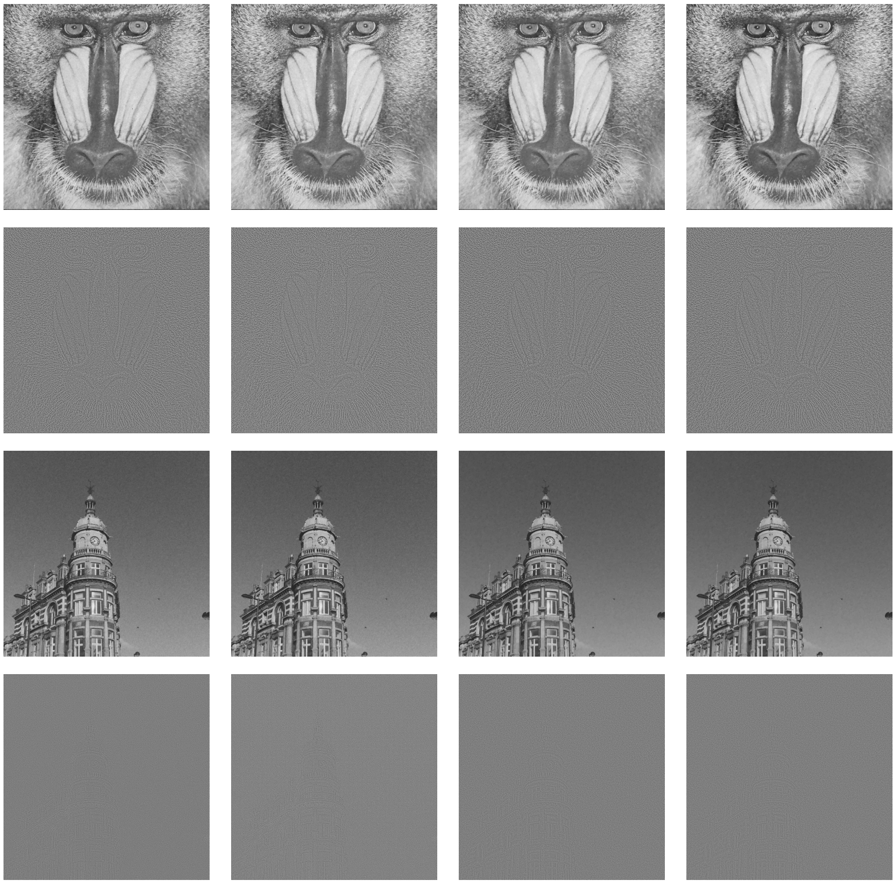
We test the aforementioned methods on eight images of different scales as shown in Figure 4. The initial value is given as $u^0 = 0$ for all methods unless otherwise specified.

The most important parameter in the multi-grid scheme is the maximum level. Thus, we compare the performance of both MMC and MML on test images with different values of maximum levels in Table 1, where the regularization parameter α is chosen as $\alpha = 10, 15, 20$, respectively. Based on the experiments, we have the following observations

- Both MML and MMC converge to similar numerical energies for different combinations of images and parameters, i.e., α and J, which means the piecewise constant functions can be substituted for the piecewise linear functions to span the subspaces for the multi-grid scheme;
- By comparing the efficiency of MML and MMC, we find out that much CPU time can be saved using piecewise constant basis functions no matter how many layers of meshes are used;
- By comparing the results of MMC/MML with one grid and multi-grid, we observe that introducing the coarse layers can greatly reduce the outer iterations. As the regularization parameter α increases, the computational costs increase and the advantages of the multi-grid structure become more significant;
- Much CPU time can be saved by increasing the maximum level from $J = 1$ to $J = 4$, while CPU time increases as the number of layers keep increasing. Therefore, we set the maximum level as $J = 4$ for both MML and MMC in the following experiments.

Next, we compare the performance of our MMC with other methods on noisy images, where the Gaussian white noises of mean 0 and variance $\sigma = 20$ are used to degrade the test images. Table 2 provides the PSNR, SSIM, CPU time, and numerical energies obtained by MMC, MML, Algorithm 1 and Algorithm 2. As can be seen, all methods converge to results with similar PSNR and numerical energies, while much CPU time is saved by our MMC method. Especially, the advantage of our multi-grid algorithm becomes more prominent for larger α, which is nearly twice faster than MML and dual Algorithm 1. Indeed, our MMC is also faster than the primal-dual Algorithm 2, which is recognized as a fast and effective algorithm for solving the Rudin-Osher-Fatemi model.

Finally, we evaluate the performance of the proposed method on images with different noise levels, i.e., white Gaussian noises of mean 0 and variance $\sigma = 20$, 30, 40, the results of which are provided in Table 3. Our MMC algorithm is robust to noise levels and regularization parameters, which gives similar PSNR values and numerical energies, but is much faster than MML. Moreover, the MMC performs consistently better than the dual and primal-dual algorithms for different noise levels. We display the denoised results and residual results of image 'Baboon' (#6) with noise level $\sigma = 20$ and image 'Building' (#8) with noise level $\sigma = 40$ in Figure 5, where similar restoration results are obtained by different methods. Besides, we record the decays of the relative errors and numerical energies of image 'Baboon' of noise level $\sigma = 20$, and image 'Building' of noise level $\sigma = 40$ in Figure 6, which clearly show that both MM methods converge to similar numerical energies.



(a) MMC (b) MML (c) Dual (d) P-D

Figure 5. Denoised results and the residual images obtained by different methods on image 'Baboon' (#6) of the noise level $\sigma = 20$ and $\alpha = 15$, and image 'Building' (#8) of noise level $\sigma = 40$ and $\alpha = 35$.

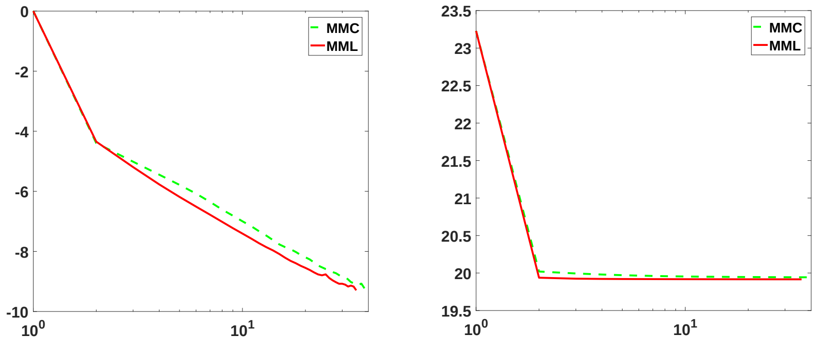
5.2. **Image deblurring.** We also test the performance of our MMC method on the deblurring problems, where K represents the blur kernels. Both the Gaussian blur of size 5×5 with standard deviation $\sigma = 5$ and motion blur of 20 pixels with an angle of $\pi/4$ are used to degrade images. We compare our MMC with the best performer of the comparison methods (PD method) on two test images 'Boat' and 'Lena' of size 512×512. The implementation details are presented as follows

1) MMC method: We set $N_c = N_w = 100$, $\epsilon_{inner} = 10^{-2}$, $\epsilon_{outer} = 10^{-4}$, and the stopping criteria of the fixed point scheme is the maximum iteration number to be 10 and $\|c^{\ell} - c^{\ell-1}\|_{\infty} < 10^{-2}$. The conjugate gradient is terminated as maximum iteration of 10 and the tolerance of 10^{-5}. Besides, $\beta = 10^{-6}$ is used for all experiments. The regularization parameter α is chosen as $\alpha = 5 \times 10^{-3}$ for both test images.

Table 2. The denoising results by MMC and MML methods for Rudin-Osher-Fatemi model with $\alpha = 10,\ 15,\ 20$ for noise level $\sigma = 20$.

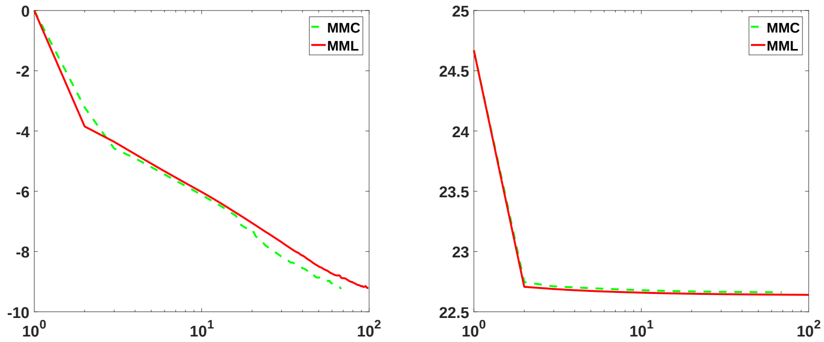
		$\alpha = 10$				$\alpha = 15$				$\alpha = 20$			
		MMC	MML	Dual	P-D	MMC	MML	Dual	P-D	MMC	MML	Dual	P-D
PSNR	#1	**28.77**	**28.77**	28.71	28.70	29.26	**29.27**	29.23	**29.27**	29.14	29.14	29.13	**29.17**
	#2	**28.43**	28.42	28.40	28.41	**28.64**	**28.64**	28.60	28.62	**27.61**	27.59	27.55	27.58
	#3	**29.73**	**29.73**	29.70	**29.73**	**30.79**	30.77	30.72	30.75	30.60	**30.63**	30.58	30.60
	#4	29.84	**29.86**	29.83	29.83	**31.14**	31.13	31.09	31.11	30.28	**30.32**	30.28	30.30
	#5	**29.18**	29.16	29.15	29.15	**29.64**	**29.64**	29.63	29.63	**29.35**	29.34	29.30	29.33
	#6	27.81	**27.82**	27.80	27.79	**27.95**	**27.95**	27.89	27.94	**26.58**	**26.58**	26.55	26.55
	#7	**30.70**	30.65	30.60	30.68	**32.45**	32.44	32.43	32.40	32.33	32.33	32.26	**32.38**
	#8	**30.43**	30.40	30.39	30.41	**32.28**	32.25	32.23	32.27	**31.27**	31.26	31.20	31.25
SSIM	#1	**0.7786**	0.7771	0.7735	0.7757	0.8511	0.8521	0.8515	**0.8523**	0.8492	0.8499	0.8491	**0.8501**
	#2	**0.7473**	0.7451	0.7376	0.7391	**0.8303**	0.8279	0.8299	0.8255	0.8243	0.8216	0.8255	**0.8299**
	#3	**0.7342**	0.7277	0.7225	0.7245	**0.8256**	0.8221	0.8242	0.8253	0.8195	0.8207	0.8217	**0.8228**
	#4	**0.7544**	0.7431	0.7419	0.7433	0.8548	**0.8562**	0.854	0.8537	0.8518	**0.8548**	0.8520	0.8521
	#5	0.7287	**0.7298**	0.7267	0.7294	**0.7649**	0.7631	0.7622	0.7641	0.7381	0.7394	0.7388	**0.7395**
	#6	**0.7953**	0.7909	0.7943	0.7952	0.8031	**0.8131**	0.8124	0.8122	0.7883	**0.7894**	0.7788	0.7793
	#7	**0.7447**	0.7412	0.7368	0.7342	0.8865	**0.8877**	0.8788	0.8791	0.8725	0.8717	0.8736	**0.8744**
	#8	**0.7386**	0.7252	0.7237	0.7241	**0.9203**	0.9107	0.9154	0.9165	0.9154	**0.9199**	0.9198	0.9192
CPU(s)	#1	**0.88**	1.53	2.41	1.40	**1.21**	3.35	4.49	2.15	**1.20**	3.61	7.61	2.71
	#2	**1.00**	1.64	1.56	1.72	**1.17**	2.01	2.81	1.97	**1.18**	3.11	9.01	2.21
	#3	**4.27**	6.94	8.26	5.62	**5.97**	9.83	14.42	8.60	**5.81**	11.28	20.67	13.94
	#4	**4.22**	6.81	8.56	6.13	**5.13**	9.13	15.01	8.11	**5.22**	12.10	21.35	11.65
	#5	**22.68**	33.19	30.12	19.73	**28.68**	48.35	54.08	29.93	**34.28**	52.13	63.59	41.57
	#6	**18.33**	23.19	31.01	20.21	**26.78**	50.14	51.35	28.86	**28.48**	56.13	66.01	39.73
	#7	**88.80**	122.73	184.93	113.37	**106.43**	192.49	463.12	161.99	**115.83**	280.32	560.32	188.35
	#8	**116.35**	133.84	190.21	121.61	**123.28**	183.17	454.35	159.73	**134.78**	244.35	555.38	198.46
Energy	#1	2.31E7	**2.29E7**	2.31E7	2.31E7	3.03E7	**3.01E7**	3.03E7	3.03E7	**3.52E7**	**3.52E7**	3.53E7	3.53E7
	#2	2.44E7	**2.43E7**	**2.43E7**	**2.43E7**	3.23E7	**3.21E7**	3.23E7	3.23E7	**3.70E7**	3.71E7	3.71E7	**3.70E7**
	#3	**1.28E8**	1.29E8	1.29E8	**1.28E8**	**2.77E8**	2.78E8	2.78E8	2.78E8	**4.66E8**	4.67E8	4.67E8	4.67E8
	#4	8.39E7	**8.37E7**	8.39E7	8.39E7	**1.08E8**	1.09E8	1.09E8	1.09E8	**1.22E8**	1.23E8	**1.22E8**	1.23E8
	#5	3.53E8	**3.51E8**	3.53E8	3.53E8	4.59E8	**4.57E8**	**4.57E8**	**4.57E8**	5.23E8	**5.21E8**	5.22E8	5.22E8
	#6	**3.71E8**	3.72E8	3.72E8	3.72E8	**4.71E8**	4.72E8	**4.71E8**	**4.71E8**	**5.97E8**	**5.97E8**	5.98E8	5.98E8
	#7	1.37E9	**1.36E9**	1.37E9	**1.36E9**	**1.75E9**	**1.75E9**	**1.75E9**	**1.75E9**	1.95E9	**1.94E9**	1.95E9	1.95E9
	#8	**1.40E9**	1.42E9	**1.40E9**	1.41E9	1.81E9	**1.80E9**	1.81E9	**1.80E9**	**1.97E9**	1.98E9	1.98E9	**1.97E9**

2) Primal-dual method: The step size is given as $\tau = 1/(2L_F)$ and $\sigma = L_F/L^2$ for the primal and dual variable, respectively, where L_F is the Lipschitz constant of $F(u) = \|Ku - f\|_2^2$ and $L^2 = \|\nabla\|^2$. The iteration process is terminated when both $\|u^{t+1} - u^t\|_2 < \epsilon\|u^{t+1}\|_2$ and $\|\boldsymbol{p}^{t+1} - \boldsymbol{p}^t\|_2 < \epsilon\|\boldsymbol{p}^{t+1}\|_2$ with $\epsilon =$



(a) #6-error

(b) #6-energy



(c) #8-error

(d) #8-energy

Figure 6. The decay of relative error and numerical energy for Baboon image (#6) of noise level $\sigma = 20$ with $\alpha = 15$ and Building image (#8) of noise level $\sigma = 40$ with $\alpha = 35$. From left to right on the first row is the relative error and numerical energy of Baboon image (#6) while on the second row are relative error and numerical energy of Building image (#8).

10^{-4}. The regularization parameter α is set to the same value as the MMC method.

Table 4 displays the PSNR, SSIM, CPU time, and numerical energies of the two methods. As illustrated, our MMC produces similar PSNR and SSIM values but saves much CPU time compared to the PD method. Besides, we also exhibit reconstructed images degraded by the motion blur in Figure 7, where similar visual results are obtained by the MMC and PD methods.

5.3. **CT reconstruction.** This subsection is devoted to the simulation of CT image reconstruction, where K in (24) denotes the forward system matrix. Two phantom images ‘Shepp-Logan’ and ‘Forbild-gen’ of size 256×256 and 512×512, are used to evaluate the performance. We use the parallel-beam geometry for both images in the experiments and set the projection numbers as $N_p = 18,\ 36$ and 72, respectively.

TABLE 3. The denoising results for Rudin-Osher-Fatemi model with different noise levels $\sigma = 20,\ 30,\ 40$.

		$\alpha = 15,\ \sigma = 20$				$\alpha = 25,\ \sigma = 30$				$\alpha = 35,\ \sigma = 40$			
		MMC	MML	Dual	P-D	MMC	MML	Dual	P-D	MMC	MML	Dual	P-D
PSNR	#1	29.26	**29.27**	29.23	**29.27**	**26.73**	**26.73**	**26.73**	**26.73**	**25.04**	**25.04**	**25.04**	**25.04**
	#2	**28.64**	**28.64**	28.60	28.62	**26.49**	26.44	26.40	26.42	**25.01**	24.95	24.96	24.96
	#3	**30.79**	30.77	30.72	30.75	**28.78**	**28.78**	**28.78**	**28.78**	**27.25**	27.21	27.21	27.21
	#4	**31.14**	31.13	31.09	31.11	**29.13**	29.11	29.11	29.12	**27.21**	**27.21**	**27.21**	**27.21**
	#5	**29.64**	**29.64**	29.63	29.63	28.73	**27.75**	**27.75**	27.74	**26.40**	26.38	26.39	**26.40**
	#6	**27.95**	**27.95**	27.89	27.94	25.44	**25.45**	25.42	24.13	23.93	**23.94**	23.92	**23.94**
	#7	**32.45**	32.44	32.43	32.40	**29.92**	29.91	29.90	29.91	**28.20**	**28.20**	**28.20**	**28.20**
	#8	**32.28**	32.25	32.23	32.27	**29.95**	29.93	29.85	29.87	**27.52**	**27.52**	27.51	27.51
SSIM	#1	0.8511	0.8521	0.8515	**0.8523**	0.8051	0.8048	0.8046	**0.8084**	0.7561	0.7563	0.7601	**0.7623**
	#2	**0.8303**	0.8279	0.8299	0.8255	0.7782	0.7755	0.7749	**0.7811**	**0.7499**	0.7476	0.7377	0.7492
	#3	**0.8256**	0.8221	0.8242	0.8253	0.7842	0.7827	0.7841	**0.7851**	**0.7555**	0.7519	0.7549	0.7537
	#4	0.8548	**0.8562**	0.8542	0.8537	**0.8046**	0.8011	0.8044	0.8012	0.7717	**0.7782**	0.7739	0.7717
	#5	**0.7649**	0.7631	0.7622	0.7641	**0.7162**	0.7144	0.7088	0.7101	**0.6772**	0.6766	0.6688	0.6768
	#6	0.8031	**0.8131**	0.8124	0.8122	0.7311	**0.7353**	0.7344	0.7349	0.6601	**0.6721**	0.6687	0.6699
	#7	0.8865	**0.8877**	0.8788	0.8791	0.8478	**0.8516**	0.8496	0.8502	0.8224	0.8236	0.8235	**0.8247**
	#8	**0.9203**	0.9107	0.9154	0.9165	**0.9002**	0.8997	0.8981	0.8994	0.8811	**0.8868**	0.8714	0.8726
CPU(s)	#1	**1.21**	3.35	4.49	2.15	**1.62**	3.88	4.35	3.99	**2.55**	5.95	5.69	5.81
	#2	**1.17**	2.01	2.81	1.97	**1.79**	3.61	7.13	4.23	**3.05**	11.78	7.13	8.68
	#3	**5.97**	9.83	14.42	8.60	**6.93**	16.06	17.14	11.86	**9.88**	37.98	17.19	15.06
	#4	**5.13**	9.13	15.01	8.11	**7.68**	15.76	19.06	8.99	**9.85**	20.74	18.99	15.04
	#5	**28.68**	48.35	54.08	29.93	**35.66**	76.67	64.43	49.30	**48.53**	190.87	78.30	56.35
	#6	**26.78**	50.14	51.35	28.86	**38.63**	79.12	78.84	58.84	**50.25**	180.62	64.30	55.44
	#7	**106.43**	192.49	463.12	161.99	**161.37**	334.45	452.12	188.14	**186.92**	491.83	443.35	269.35
	#8	**123.28**	183.17	454.35	159.73	**164.28**	303.28	443.18	191.18	**208.15**	482.78	482.06	275.05
Energy	#1	3.03E7	**3.01E7**	3.03E7	3.03E7	**6.56E7**	6.57E7	6.57E7	**6.56E7**	1.18E8	**1.17E8**	1.18E8	**1.17E8**
	#2	3.23E7	**3.21E7**	3.23E7	3.23E7	**6.84E7**	6.85E7	6.85E7	**6.84E7**	**1.15E8**	1.16E8	1.16E8	**1.15E8**
	#3	**2.77E8**	2.78E8	2.78E8	2.78E8	2.87E8	2.87E8	**2.86E8**	**2.86E8**	**4.35E8**	4.36E8	4.36E8	**4.35E8**
	#4	**1.08E8**	1.09E8	1.09E8	1.09E8	**2.42E8**	2.43E8	2.43E8	2.43E8	4.30E8	**4.29E8**	4.31E8	4.31E8
	#5	4.59E8	**4.57E8**	**4.57E8**	**4.57E8**	**1.01E9**	1.02E9	1.02E9	**1.01E9**	**1.77E9**	1.78E9	1.78E9	**1.77E9**
	#6	4.72E8	**4.71E8**	**4.71E8**	**4.71E8**	**1.08E9**	1.09E9	1.09E9	**1.08E9**	**1.87E9**	1.88E9	1.88E9	**1.87E9**
	#7	**1.75E9**	**1.75E9**	**1.75E9**	**1.75E9**	3.94E9	**3.93E9**	**3.93E9**	**3.93E9**	6.95E9	**6.94E9**	6.95E9	**6.94E9**
	#8	1.81E9	**1.80E9**	1.81E9	**1.80E9**	**3.98E9**	3.99E9	3.99E9	3.99E9	**7.31E9**	7.32E9	**7.31E9**	**7.31E9**

The implementation details for comparison are described as follows

1) MMC method: We set $N_c = N_w = 100$, $\epsilon_{inner} = 10^{-2}$, $\epsilon_{outer} = 10^{-4}$, and the stopping criteria of the fixed point scheme is maximum iteration number of 10 and $\|c^\ell - c^{\ell-1}\|_\infty < 10^{-2}$. The conjugate gradient is terminated with

TABLE 4. PSNR, SSIM values and CPUs for deblurring with different kernel.

	Image	Gaussian Blur				Motion Blur			
		CPU	PSNR	SSIM	Energy	CPU	PSNR	SSIM	Energy
MMC	Lena	**8.63**	**34.85**	0.9207	1.4840	**14.34**	**35.41**	**0.9401**	1.5673
PD		16.22	34.82	**0.9234**	**1.4713**	19.41	35.36	0.9382	**1.5610**
MMC	Boat	**16.69**	**33.83**	0.9204	**1.9612**	**18.98**	**33.43**	**0.9178**	**1.9549**
PD		21.22	33.86	**0.9222**	1.9623	29.41	33.26	0.9166	1.9568

(a) Clean image (b) Blurry image (c) MMC (d) PD

FIGURE 7. Image deburring results obtained by our MMC and PD method on 'Boat' and 'Lena' with motion blur.

maximum iteration 10 and error tolerance 10^{-5}. Besides, $\beta = 10^{-6}$ is fixed for all experiments. The regularization parameter α is chosen as $\alpha = 3.5 \times 10^{-5}$ for projection number $N_p = 18$ and $\alpha = 2 \times 10^{-5}$ for $N_p = 36$ and 72.

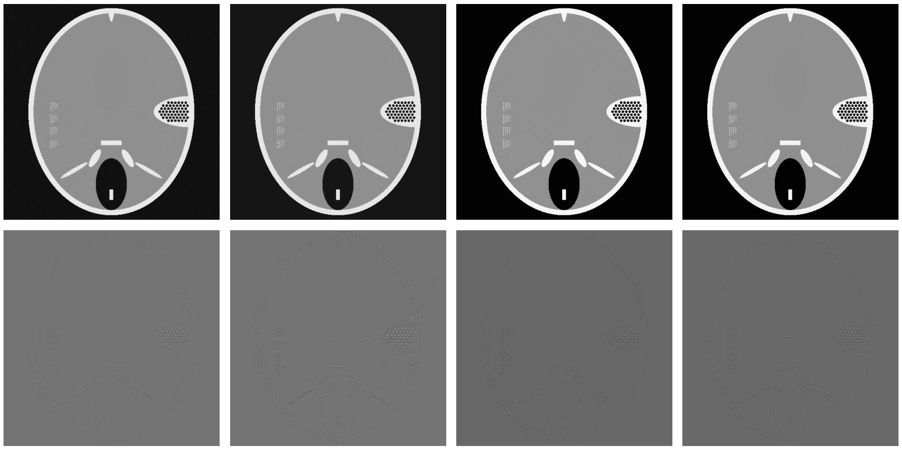
2) Primal-dual method: The step size is given as $\tau = 1/(4L_F)$ and $\sigma = L_F/L^2$ for the primal and dual variable [8], where L_F is the Lipschitz constant of the fidelity term $F(u) = \|Ku - f\|_2^2$ and $L^2 = \|\nabla\|^2$. The iteration is terminated when both of the following criteria are met: $\|u^{t+1} - u^t\|_2 < \epsilon\|u^{t+1}\|_2$ and $\|\boldsymbol{p}^{t+1} - \boldsymbol{p}^t\|_2 < \epsilon\|\boldsymbol{p}^{t+1}\|_2$ with $\epsilon = 10^{-4}$. The regularization parameter α is set the same as our MMC algorithm.

Figure 8 displays both the reconstructed images and residual images from the sinogram observation with $N_p = 36$ and 72 of the 'Forbild-gen' phantom. As shown, both methods present similar visual results for both sparse reconstruction problems. The comparisons results of PSNR, SSIM, CPU time and numerical energies are recorded in Table 5, where a noticeable advantage of our MMC method can be observed such that much CPU time has been saved for both test images.

5.4. **MRI reconstruction.** We also test the performance of our MMC method on the under-sampled MRI reconstruction problems, where $K = \mathcal{P}\mathcal{F}$ is a composite operator of the selection operator $\mathcal{P}$ and Fourier transform $\mathcal{F}$. We use brain phantom of size 256×256 as the test image and apply both radial and random

TABLE 5. PSNR values and CPUs for CT reconstruction with different projection numbers of MMC and PD method.

	Image	N_p	18				36				72			
		Size	CPU(s)	PSNR	SSIM	Energy	CPU(s)	PSNR	SSIM	Energy	CPU(s)	PSNR	SSIM	Energy
MMC	Shepp	256	**15.64**	**39.28**	**0.9908**	**0.0508**	**20.01**	50.19	**0.9995**	**0.0292**	**33.21**	**58.30**	**0.9998**	**0.0292**
PD			27.59	39.16	0.9901	**0.0508**	59.33	**50.37**	**0.9995**	**0.0292**	112.04	58.21	**0.9998**	**0.0292**
MMC	Fobild	256	**21.33**	**27.96**	**0.9508**	**0.0595**	**24.15**	**37.66**	**0.9838**	**0.0421**	**40.21**	45.06	**0.9918**	**0.0423**
PD			25.02	27.92	0.9500	0.0600	71.21	37.17	**0.9828**	**0.0421**	140.10	**45.16**	0.9917	0.0425
MMC	Shepp	512	**96.62**	**35.79**	**0.9881**	**0.0991**	**103.54**	**43.22**	**0.9911**	0.0587	**129.91**	**50.60**	**0.9995**	**0.0585**
PD			176.86	35.72	0.9879	**0.0991**	204.57	43.21	**0.9911**	**0.0585**	370.14	50.51	**0.9995**	**0.0585**
MMC	Fobild	512	**130.01**	**31.29**	**0.9726**	**0.1501**	**153.52**	**37.90**	**0.9899**	**0.1557**	**179.58**	**48.88**	**0.9925**	**0.0861**
PD			187.02	31.23	**0.9726**	**0.1501**	248.01	37.86	**0.9896**	**0.1557**	457.51	48.83	**0.9925**	**0.0861**



(a) $N_p = 36$, MMC (b) $N_p = 36$, PD (c) $N_p = 72$, MMC (d) $N_p = 72$, PD

FIGURE 8. Both CT reconstruction results (Row one) and residual images (Row two) obtained by MMC and PD method for 'Forbilgen' phantom with $N_p = 36$ and 72.

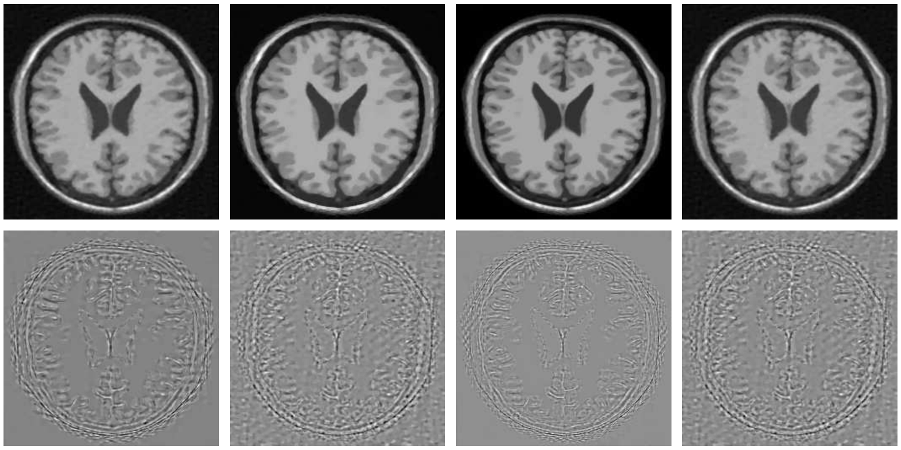
sampling patterns to generate the under-sampled data. The numbers of sampling lines are set as $N_s = 30$ and 50, which are of sampling rate 12.65% and 20.06%, respectively. Similarly, we compare the performance of our MMC with PD method, the implementation details of which are presented as follows

1) MMC method: We set $N_c = N_w = 100$, $\epsilon_{inner} = 10^{-2}$, $\epsilon_{outer} = 10^{-4}$, and the stopping criteria of the fixed point scheme is the maximum iteration number to be 10 and $\|c^\ell - c^{\ell-1}\|_\infty < 10^{-2}$. The conjugate gradient is terminated as maximum iteration of 10 and the tolerance of 10^{-5}. Besides, $\beta = 10^{-6}$ is used for all experiments. The regularization parameter α is 5×10^{-3} for both projection numbers $N_s = 30$ and 50.
2) Primal-dual method: The step size is given as $\tau = 1/(2L_F)$ and $\sigma = L_F/L^2$ for the primal and dual variable, respectively, where L_F is the Lipschitz constant of $F(u) = \|\mathcal{P}\mathcal{F}u - f\|_2^2$ and $L = \|\nabla\|$. The iteration process is terminated when both $\|u^{t+1} - u^t\|_2 < \epsilon\|u^{t+1}\|_2$ and $\|\boldsymbol{p}^{t+1} - \boldsymbol{p}^t\|_2 < \epsilon\|\boldsymbol{p}^{t+1}\|_2$ with $\epsilon = 10^{-4}$. The regularization parameter α is set to the same value as our MMC method.

Table 6 displays the PSNR, SSIM, CPU time, and numerical energies of the two methods. As illustrated, both MMC and PD algorithm converge to similar energy values. More importantly, we can find that much computational cost is saved by our MMC method due to the multi-grid structure and parallel computation. Besides, we also exhibit reconstructed images and residual images in Figure 9, where less structural information is presented in the residual images obtained by our MMC method.

Table 6. PSNR values and CPUs for MRI reconstruction with different numbers of sampling lines for the MMC and PD method.

	N_s	30				50			
	Pattern	CPU	PSNR	SSIM	Energy	CPU	PSNR	SSIM	Energy
MMC	Radial	**4.78**	**32.15**	**0.9711**	**1.8840**	**10.19**	**38.41**	**0.9847**	**1.1673**
PD		13.22	32.08	0.9701	**1.8840**	19.41	38.36	0.9837	1.1675
MMC	Random	**5.84**	**27.03**	0.9500	2.0915	**11.98**	**35.73**	**0.9812**	2.0070
PD		15.34	27.01	**0.9504**	**2.0913**	25.65	35.72	**0.9812**	**2.0070**



(a) $N_s = 30$, MMC (b) $N_s = 30$, PD (c) $N_s = 50$, MMC (d) $N_s = 50$, PD

Figure 9. MRI reconstruction results (Row one) and residual images (Row two) obtained by our MMC and PD method on the brain image with radial sampling patterns and sampling lines $N_s = 30$ and 50.

6. **Conclusion.**

We proposed a fast and efficient multi-grid method for TV minimization problems, which is built up with the piecewise constant basis functions over the multi-grid structure. In the new formulation, the computation of the TV regularization occurs on the boundaries of small-scale minimization problems, which means the computational complexity can be greatly reduced. Furthermore, we implemented the non-overlapping domain decomposition method on each layer to increase the parallelism. Numerical results on image restoration and medical image reconstruction demonstrate that the proposed multi-grid method is of high efficiency and wide-ranging application potential.

REFERENCES

[1] R. Acar and C. R. Vogel, Analysis of bounded variation penalty methods for ill-posed problems, *Inverse Problems*, **10** (1994), 1217–1229.

[2] S. T. Acton, Multigrid anisotropic diffusion, *IEEE Transactions on Image Processing: A Publication of the IEEE Signal Processing Society*, **7** (1998), 280–291.

[3] X. Bresson and T. F. Chan, Fast dual minimization of the vectorial total variation norm and applications to color image processing, *Inverse Probl. Imaging*, **2** (2008), 455–484.

[4] C. Brito-Loeza and K. Chen, On high-order denoising models and fast algorithms for vector-valued images, *IEEE Trans. Image Process.*, **19** (2010), 1518–1527.

[5] A. Chambolle, An algorithm for total variation minimization and applications, *J. Math. Imaging Vision*, **20** (2004), 89–97.

[6] A. Chambolle and T. Pock, A first-order primal-dual algorithm for convex problems with applications to imaging, *J. Math. Imaging Vision*, **40** (2011), 120–145.

[7] A. Chambolle and T. Pock, An introduction to continuous optimization for imaging, *Acta Numer.*, **25** (2016), 161–319.

[8] A. Chambolle and T. Pock, On the ergodic convergence rates of a first-order primal-dual algorithm, *Math. Program.*, **159** (2016), 253–287.

[9] R. H. Chan and K. Chen, A multilevel algorithm for simultaneously denoising and deblurring images, *SIAM J. Sci. Comput.*, **32** (2010), 1043–1063.

[10] T. F. Chan and K. Chen, On a nonlinear multigrid algorithm with primal relaxation for the image total variation minimisation, *Numer. Algorithms*, **41** (2006), 387–411.

[11] T. F. Chan and K. Chen, An optimization based multilevel algorithm for total variation image denoising, *Multiscale Model. Simul.*, **5** (2006), 615–645.

[12] T. F. Chan and J. Shen, Mathematical models for local nontexture inpaintings, *SIAM J. Appl. Math.*, **62** (2002), 1019–1043.

[13] T. F. Chan and L. A. Vese, Active contours without edges, *IEEE Transactions on Image Processing*, **10** (2001), 266–277.

[14] H. Chang, X.-C. Tai, L.-L. Wang and D. Yang, Convergence rate of overlapping domain decomposition methods for the Rudin-Osher-Fatemi model based on a dual formulation, *SIAM J. Imaging Sci.*, **8** (2015), 564–591.

[15] R. Chen, J. Huang and X.-C. Cai, A parallel domain decomposition algorithm for large scale image denoising, *Inverse Probl. Imaging*, **13** (2019), 1259–1282.

[16] K. Chen and J. Savage, An accelerated algebraic multigrid algorithm for total-variation denoising, *BIT*, **47** (2007), 277–296.

[17] K. Chen and X.-C. Tai, A nonlinear multigrid method for total variation minimization from image restoration, *J. Sci. Comput.*, **33** (2007), 115–138.

[18] Y. Duan, H. Chang and X.-C. Tai, Convergent non-overlapping domain decomposition methods for variational image segmentation, *J. Sci. Comput.*, **69** (2016), 532–555.

[19] E. Esser, X. Zhang and T. F. Chan, A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science, *SIAM J. Imaging Sci.*, **3** (2010), 1015–1046.

[20] D. Goldfarb and W. Yin, Parametric maximum flow algorithms for fast total variation minimization, *SIAM J. Sci. Comput.*, **31** (2009), 3712–3743.

[21] T. Goldstein and S. Osher, The split Bregman method for L_1-regularized problems, *SIAM J. Imaging Sci.*, **2** (2009), 323–343.

[22] Y. Gu, L.-L. Wang and X.-C. Tai, A direct approach toward global minimization for multiphase labeling and segmentation problems, *IEEE Trans. Image Process.*, **21** (2012), 2399–2411.

[23] M. Hintermüller and A. Langer, Non-overlapping domain decomposition methods for dual total variation based image denoising, *J. Sci. Comput.*, **62** (2015), 456–481.

[24] A. Langer and F. Gaspoz, Overlapping domain decomposition methods for total variation denoising, *SIAM J. Numer. Anal.*, **57** (2019), 1411–1444.

[25] C.-O. Lee and C. Nam, Primal domain decomposition methods for the total variation minimization, based on dual decomposition, *SIAM J. Sci. Comput.*, **39** (2017), B403–B423.

[26] C.-O. Lee, C. Nam and J. Park, Domain decomposition methods using dual conversion for the total variation minimization with L^1 fidelity term, *J. Sci. Comput.*, **78** (2019), 951–970.

[27] T. Lu, P. Neittaanmäki and X.-C. Tai, A parallel splitting up method and its application to Navier-Stokes equations, *Appl. Math. Lett.*, **4** (1991), 25–29.

[28] A. Marquina and S. Osher, Explicit algorithms for a new time dependent model based on level set motion for nonlinear deblurring and noise removal, *SIAM J. Sci. Comput.*, **22** (2000), 387–405.

[29] L. I. Rudin, S. Osher and E. Fatemi, Nonlinear total variation based noise removal algorithms, *Phys. D*, **60** (1992), 259–268.

[30] J. Savage and K. Chen, An improved and accelerated non-linear multigrid method for total-variation denoising, *Int. J. Comput. Math.*, **82** (2005), 1001–1015.

[31] J. Savage and K. Chen, An improved and accelerated non-linear multigrid method for total-variation denoising, *Int. J. Comput. Math.*, **82** (2005), 1001–1015.

[32] E. Y. Sidky, J. H. Jørgensen and X. Pan, Convex optimization problem prototyping for image reconstruction in computed tomography with the Chambolle-Pock algorithm, *Physics in Medicine & Biology*, **57** (2012), 3065.

[33] E. Y. Sidky and X. Pan, Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization, *Physics in Medicine & Biology*, **53** (2008), 4777.

[34] X.-C. Tai and Y. Duan, Domain decomposition methods with graph cuts algorithms for image segmentation., *Int. J. Numer. Anal. Model.*, **8** (2011), 137–155.

[35] X.-C. Tai and J. Xu, Global and uniform convergence of subspace correction methods for some convex optimization problems, *Math. Comp.*, **71** (2002), 105–124.

[36] C. R. Vogel and M. E. Oman, Iterative methods for total variation denoising, *SIAM J. Sci. Comput.*, **17** (1996), 227–238.

[37] C. R. Vogel and M. E. Oman, Fast, robust total variation-based reconstruction of noisy, blurred images, *IEEE Trans. Image Process.*, **7** (1998), 813–824.

[38] Y. Wang, J. Yang, W. Yin and Y. Zhang, A new alternating minimization algorithm for total variation image reconstruction, *SIAM J. Imaging Sci.*, **1** (2008), 248–272.

[39] C. Wu and X.-C. Tai, Augmented Lagrangian method, dual methods, and split bregman iteration for ROF, vectorial TV, and high order models, *SIAM J. Imaging Sci.*, **3** (2010), 300–339.

[40] J. Xu, Iterative methods by space decomposition and subspace correction, *SIAM Rev.*, **34** (1992), 581–613.

[41] J. Xu, H. B. Chang and J. Qin, Domain decomposition method for image deblurring, *J. Comput. Appl. Math.*, **271** (2014), 401–414.

[42] J. Xu, X.-C. Tai and L.-L. Wang, A two-level domain decomposition method for image restoration, *Inverse Probl. Imaging*, **4** (2010), 523–545.

Received August 2020; revised February 2021.

E-mail address: zzw7548@163.com
E-mail address: 2018233023@tju.edu.cn
E-mail address: yuping.duan@tju.edu.cn
E-mail address:
E-mail address: