Fast Multilevel Algorithm for a Minimization Problem in Impulse Noise Removal

Raymond H. Chan* and Ke Chen†

September 25, 2007

Abstract

An effective 2-phase method for removing impulse noise was recently proposed. Its phase 1 identifies noisy pixel candidates by using median-type filters. Then in phase 2, it restores only the noisy pixel candidates by some variational methods. The resulting method can handle salt-and-pepper noise and random-valued impulse noise of noise level as high as 90% and 60% respectively. The aim of this paper is to generalize a fast multilevel method for Gaussian denoising to solving the minimization problem arising in phase 2 of the 2-phase method. The multilevel algorithm gives better images than standard optimization method such as the Newton method or conjugate gradient method. Also it can handle more general regularization functionals than the smooth ones previously considered. Supporting numerical experiments on 2D gray scale images are presented.

AMS subject class: 68U10, 65F10, 65K10.

Keywords: Image restoration, impulse noise, regularization, multilevel methods.

1 Introduction

Image denoising is one of the basic problems in image processing [1, 9, 19, 20]: given an observed image $z \in \mathbb{R}^{n\times n}$, restore a 'quality' image $u \in \mathbb{R}^{n\times n}$ such that $z = u+\eta$ with η being some noise matrix. Here by 'quality', we mean that the problem of finding u from z is a typical inverse problem which does not have a unique solution without regularization. How to model η properly depends on the context in which the given image z is gathered. For images contaminated by environments or a transmission process, all pixels contain noise but the whole image is still vaguely 'visible' to the human eye. Then the Gaussian noise model for η is commonly used [19, 20]; we assume that η is sampled from a Gaussian distribution with zero mean and some

*Department of Mathematics, Chinese University of Hong Kong, Shatin, Hong Kong SAR, China. Email: rchan@math.cuhk.edu.hk. Web: http://www.math.cuhk.edu.hk/∼rchan. Research supported in part by HKRGC grant no. 400405, and CUHK DAG grant no. 2060257

†Department of Mathematical Sciences, University of Liverpool, Peach Street, Liverpool L69 7ZL, UK. Email: k.chen@liverpool.ac.uk. Web: http://www.liv.ac.uk/∼cmchenke

variance which may be estimated from z. For other noisy images generated by imperfect imaging equipment e.g. malfunctioning sensors and faulty memory, the impulse noise model for z appears to be more appropriate [10, 14]. Here although the underlying image may not be visible to human eyes, the belief is that some pixels do contain the true values and the others contain completely incorrect values, with both locations being unknown.

Recently an effective 2-phase method for removing impulse noise was proposed [4, 6]. In phase 1, it tries to identify noisy pixel candidates by using some median-type filters; see [10, 14, 15, 18] and the references therein. Then in phase 2, it restores only the noisy pixel candidates by variational methods. It is similar to doing an inpainting on the noisy pixel candidates. By using the functional proposed in [14], the resulting method can handle salt-and-pepper noise and random-valued impulse noise of noise level as high as 90% and 60% respectively, see [4, 6, 12]. The main difficulty in this 2-phase method is in solving the optimization problem that arises in the second phase. The minimization functional consists of an ℓ_1 data fitting term and either a smooth or non-smooth regularization term, see [14, 6, 4].

In [5, 1], we have shown that in phase 2, there is no need to consider the data fitting term as the data are fitted exactly for non-noisy candidates. It remains only to minimize the smooth or non-smooth regularization functional over the noisy candidate set obtained from phase 1. In this paper, we will develop a multilevel acceleration method for this optimization problem which is based on the method in [7] for Gaussian noise, and is more robust and reliable even for non-smooth regularization functional such as the total-variation norm model [19].

We will use the following notations. Let $\mathcal{A}$ denote the index set of all pixels i.e. $\mathcal{A} = \{(i,j) \mid i = 1,\ldots,n; j = 1,\ldots,n\}$ and $\mathcal{N}$ denote the index set of pixels that are found noisy by a filtering method in phase 1. The remaining pixels are denoted by $\mathcal{N}^c = \mathcal{A}\backslash\mathcal{N}$, and we will call them simply as correct pixels. For any noisy candidate pixel $(i,j) \in \mathcal{N}$, denote by $\mathcal{V}_{i,j}$ the set of its neighbors (not including itself). Then the splitting $\mathcal{V}_{i,j} = (\mathcal{V}_{i,j} \cap \mathcal{N}) \bigcup (\mathcal{V}_{i,j} \cap \mathcal{N}^c)$ will separate correct pixels from the noisy candidates.

In the second phase of the 2-phase method, we shall restore the pixels in $\mathcal{N}$ by solving the following minimization problem [5, 1]:

$$\min_{u_{i,j}\in\mathbb{R},(i,j)\in\mathcal{N}} F(u), \quad F(u) = \sum_{(i,j)\in\mathcal{N}} \left(\sum_{(m,r)\in\mathcal{V}_{i,j}\cap\mathcal{N}^c} 2\phi(u_{i,j} - z_{m,r}) + \sum_{(m,r)\in\mathcal{V}_{i,j}\cap\mathcal{N}} \phi(u_{i,j} - u_{m,r}) \right), \tag{1}$$

where ϕ is an even edge-preserving potential function. We emphasize that since data-fitting is done exactly on $\mathcal{N}^c$, there is only the regularization term in formulation (1) and hence no regularization parameter is needed here. Examples of the above ϕ are:

$$\phi(t) = |t| \tag{2}$$

$$\phi(t) = \sqrt{t^2 + \beta}, \quad \beta > 0, \tag{3}$$

$$\phi(t) = |t|^\gamma, \qquad 1 < \gamma \leq 2. \tag{4}$$

Here when (2) is used, one normally approximates it by (3) with a small β as a regularized version — consequently one has to address the sensitivity of numerical techniques with respect to the choice of β.

For smooth $\phi(t)$, e.g. (3) and (4) above, one can solve (1) by some standard optimization methods. In fact, Newton's method with continuation and conjugate gradient method were tried in [5] and [1] respectively. Readers familiar with the Gaussian noise removal [19] can connect (2) to the variational semi-norm [16]

$$\|u\|^* = \int_\Omega (|u_x| + |u_y|)$$

where $\Omega = [0,1]^2$ is the continuous image domain. This norm is non-rotationally-invariant, but one may immediately relate it to the well-known rotationally-invariant total variation (TV) semi-norm [19, 9, 20]:

$$\|u\|_{TV} = \int_\Omega \sqrt{u_x^2 + u_y^2}.$$

Using the TV semi-norm, we can suggest a similar restoration model to (1):

$$\min_{u_{i,j} \in \mathbb{R}, (i,j) \in \mathcal{N}} F_{TV}(u), \qquad F_{TV}(u) = \sum_{(i,j) \in \mathcal{N}} \sqrt{(u_{i,j} - u_{i+1,j})^2 + (u_{i,j} - u_{i,j+1})^2}, \tag{5}$$

where we take as zero those differences involving $u_{i,n+1}, u_{n+1,j}$ for all i, j. Here $u_{m,r} = z_{m,r}$ for all $(m,r) \in \mathcal{N}^c$ as these indices correspond to the correct pixels.

Our task here is to accelerate the solution procedure of variational models in phase 2. That is, we shall discuss how to solve the minimization problems (1) and (5). For smooth $\phi(t)$, one can solve the Euler-Lagrange equation of (1) as we did in [5, 1]. Standard multigrid [2, 11] is difficult to use on the equation since we are solving it on the irregular grid points $\mathcal{N}$ and coarsening will be a problem. The method that we are going to use is a multilevel method

that solve the minimization problems (1) and (5) directly and has been shown to work well for Gaussian noise removal [7]. Since minimization is not done for the whole index set $\mathcal{A}$, but on $\mathcal{N}$, special modification is needed. Problem (1) with the choice (3) will be considered as the main task while discussions on problem (1) with (2) and problem (5), although not done before because of the non-smoothness of the regularization functionals, will be supplementary as the multilevel adaption is the same.

The plan is to review our recently proposed multilevel method for the Gaussian noise removal [7] in Section 2. Section 3 presents details of implementation of a multilevel method for model (1). Numerical experiments are reported in Section 4, where we shall use the new method first as a combined method with the Newton continuation method [5] and then as a standalone method. The merits with using (5) are also considered and highlighted.

2 Review of a multilevel method for optimization

We now briefly review the multilevel method proposed in [7] for removing Gaussian noise. One nice advantage about the method is that it can be applied to non-smooth functionals as it does not require their derivatives. We illustrate the method in solving standard TV model [19]:

$$\min_u J(u), \qquad J(u) = \int_\Omega \left(\overline{\alpha} \sqrt{u_x^2 + u_y^2} + \frac{1}{2}(u - z)^2 \right),$$

which is discretized to give rise to the optimization problem (given $z \in \mathbb{R}^{n\times n}$ as before)

$$\min_{u\in\mathbb{R}^{n\times n}} J(u), \quad J(u) = \alpha \sum_{i=1}^{n-1}\sum_{j=1}^{n-1} \sqrt{(u_{i,j} - u_{i,j+1})^2 + (u_{i,j} - u_{i+1,j})^2} + \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(u_{i,j} - z_{i,j})^2, \quad (6)$$

with $\alpha = \overline{\alpha}/h$ and $h = 1/(n-1)$. For simplicity, we shall assume $n = 2^L$. Let the standard coarsening be used giving rise to $L+1$ levels $k = 1$ (finest), 2, ..., L, $L+1$ (coarsest). Denote the dimension of level k by $\tau_k \times \tau_k$ with $\tau_k = n/2^{k-1}$.

As a prelude to multilevel methods, consider the minimization of (6) by the coordinate descent method on the finest level 1:

$$\begin{cases} \text{Given} & u^{(0)} = (u_{i,j}^{(0)}) = (z_{i,j}) \text{ with } l = 0, \\ \text{Solve} & u_{i,j}^{(l)} = \text{argmin}_{u_{i,j}\in\mathbb{R}} J^{\text{loc}}(u_{i,j}) \text{ for } i,j = 1,2,\ldots,n \\ \text{Set} & u^{(l+1)} = (u_{i,j}^{(l)}) \text{ and repeat the above step with } l = l+1 \\ & \text{until a prescribed stopping step on } l, \end{cases} \quad (7)$$

where

$$J^{\rm loc}(u_{i,j}) = \alpha\Big[\sqrt{(u_{i,j}-u^{(l)}_{i+1,j})^2+(u_{i,j}-u^{(l)}_{i,j+1})^2}\ +\sqrt{(u_{i,j}-u^{(l)}_{i-1,j})^2+(u^{(l)}_{i-1,j}-u^{(l)}_{i-1,j+1})^2} \\ +\sqrt{(u_{i,j}-u^{(l)}_{i,j-1})^2+(u^{(l)}_{i,j-1}-u^{(l)}_{i+1,j-1})^2}\ \Big]\ +\frac{1}{2}(u_{i,j}-z_{i,j})^2. \tag{8}$$

Due to Neumann's condition for the continuous variable u, all difference terms involving indices in subscripts larger than n are set to zero. Note that each subproblem in (7) is only one dimensional.

To introduce the multilevel algorithm, we rewrite (7) in an equivalent form:

$$\begin{cases} \text{Given} & u^{(0)} = (u^{(0)}_{i,j}) = (z_{i,j}) \text{ with } l = 0, \\ \text{Solve} & \hat{c} = \text{argmin}_{c\in\mathbb{R}} J^{\rm loc}(u^{(l)}_{i,j}+c),\ \text{ set } u^{(l)}_{i,j} = u^{(l)}_{i,j} + \hat{c} \text{ for } i,j = 1,2,\ldots,n \\ \text{Set} & u^{(l+1)} = (u^{(l)}_{i,j}) \text{ and repeat the above step with } l = l+1 \\ & \text{until a prescribed stopping step on } l. \end{cases} \tag{9}$$

Here each subproblem can be interpreted as finding the best correction constant $\hat{c}$ at the current approximate $u^{(l)}_{i,j}$ on level 1. Likewise one may consider a 2×2 block of pixels with pixel values denoted by the current approximate $\widetilde{u}$. Our multilevel method for $k = 2$ is to look for the best correction constant to update this block so that the underlying merit functional (relating to all four pixels) achieves a local minimum. One sees that this idea operates on level 2. If we repeat the idea with larger blocks, we arrive at levels 3, 4 with respective 4×4 and 8×8 blocks.

If we write down the above idea in formulae, it may appear complicated but the idea is simple. On level k, set $b = 2^{k-1},\ k_1 = (i-1)b+1,\ k_2 = ib,\ \ell_1 = (j-1)b+1,\ \ell_2 = jb$. Then the $(i,j)^{th}$ computational block (stencil) involving the single constant $c_{i,j}$ on level k can be depicted in terms of pixels of level 1 as follows

$$\begin{array}{c|c|c|c} \vdots & \vdots & \cdots & \vdots \\ \widetilde{u}_{k_1-1,\ell_2+1}+c_{i-1,j+1} & \widetilde{u}_{k_1,\ell_2+1}+c_{i,j+1} & \cdots & \widetilde{u}_{k_2,\ell_2+1}+c_{i,j+1} \\ \hline \widetilde{u}_{k_1-1,\ell_2}+c_{i-1,j} & \widetilde{u}_{k_1,\ell_2}+c_{i,j} & \cdots & \widetilde{u}_{k_2,\ell_2}+c_{i,j} \\ \cdots & \vdots & \cdots & \vdots \\ \widetilde{u}_{k_1-1,\ell_1}+c_{i-1,j} & \widetilde{u}_{k_1,\ell_1}+c_{i,j} & \cdots & \widetilde{u}_{k_2,\ell_1}+c_{i,j} \\ \hline \widetilde{u}_{k_1-1,\ell_1-1}+c_{i-1,j-1} & \widetilde{u}_{k_1,\ell_1-1}+c_{i,j-1} & \cdots & \widetilde{u}_{k_2,\ell_1-1}+c_{i,j-1} \\ \vdots & \vdots & \cdots & \vdots \end{array} \qquad \begin{array}{c} \vdots \\ \widetilde{u}_{k_2+1,\ell_2+1}+c_{i+1,j+1} \\ \widetilde{u}_{k_2+1,\ell_2}+c_{i+1,j} \\ \cdots \\ \widetilde{u}_{k_2+1,\ell_1}+c_{i+1,j} \\ \widetilde{u}_{k_2+1,\ell_1-1}+c_{i+1,j-1} \\ \vdots \end{array} \tag{10}$$

Clearly there is only one unknown constant $c_{i,j}$ and we shall obtain a one-dimensional subproblem. After some algebraic manipulation [7, 8], we find that the local minimization problem $\min_{c_{i,j}} J(\widetilde{u}+P_k c_{i,j})$ (with P_k an interpolation operator distributing $c_{i,j}$ to a $b\times b$ block on level

k as illustrated above) is equivalent to problem $\min_{c_{i,j}} G(c_{i,j})$, where

$$\begin{aligned} G(c_{i,j}) &= \alpha \sum_{\ell=\ell_1}^{\ell_2} \sqrt{(c_{i,j} - h_{k_1-1,\ell})^2 + v_{k_1-1,\ell}^2} + \alpha \sum_{m=k_1}^{k_2-1} \sqrt{(c_{i,j} - v_{m,\ell_2})^2 + h_{m,\ell_2}^2} + \\ & \alpha \sum_{\ell=\ell_1}^{\ell_2-1} \sqrt{(c_{i,j} - h_{k_2,\ell})^2 + v_{k_2,\ell}^2} + \alpha \sum_{m=k_1}^{k_2} \sqrt{(c_{i,j} - v_{m,\ell_1-1})^2 + v_{m,\ell_1-1}^2} + \\ & \alpha\sqrt{2}\sqrt{(c_{i,j} - \overline{v}_{k_2,\ell_2})^2 + \overline{h}_{k_2,\ell_2}^2} + \frac{b^2}{2}(c_{i,j} - \widetilde{w}_{i,j})^2, \end{aligned} \tag{11}$$

and

$$\begin{cases} \widetilde{z}_{m,\ell} = z_{m,\ell} - \widetilde{u}_{m,\ell}, \quad \widetilde{w}_{i,j} = \text{mean}\Big(\widetilde{z}(k_1 : k_2, \ell_1 : \ell_2)\Big) = \dfrac{1}{b^2} \displaystyle\sum_{m=k_1}^{k_2} \sum_{\ell=\ell_1}^{\ell_2} \widetilde{z}(m,\ell), \\ \widetilde{v}_{m,\ell} = \widetilde{u}_{m,\ell+1} - \widetilde{u}_{k,\ell}, \overline{v}_{k_2,\ell_2} = \dfrac{v_{k_2,\ell_2} + h_{k_2,\ell_2}}{2}, \widetilde{h}_{m,\ell} = \widetilde{u}_{m+1,\ell} - \widetilde{u}_{m,\ell}, \overline{h}_{k_2,\ell_2} = \dfrac{v_{k_2,\ell_2} - h_{k_2,\ell_2}}{2}. \end{cases} \tag{12}$$

The solution of the above 1D minimization problem defines the updated solution of $u = \widetilde{u} + P_k c_{i,j}$. Then we obtain a multilevel method if we cycle through all levels and all blocks on each level.

Two remarks are due here. Firstly would the derived multilevel algorithm converge? The answer is no, if the functional J is non-smooth. However, in [7], we found that the wrongly converged solution is only incorrect near flat patches of the solution. The idea of detecting such flat patches during iteration and incorporating new local minimizations based on the patches was suggested in [7]. Essentially we implement a new coarse level. Secondly, how would one solve the one-dimensional minimization problem? Our experience suggests either a fixed-point based Richardson iteration or the Newton method. On the coarsest level, the TV term is unchanged by adding c so the problem has an exact solution. To avoid the gradient becoming zero on other levels, we need a regularizing parameter δ [7] which does not influence the final convergence as long as it is small (e.g. 10^{-20}). Here the solution of each local minimization problem is only needed to be approximate as with smoothing steps of a multigrid method for an operator equation. One might question the advantage of solving (6) this way or ask: why cannot one solve a regularized version of (6) at the very beginning? The reason is that with small but nonzero δ the local patches (of the true solution) are smoothed out and are less easy to be detected in order to speed up the convergence by a specially designed coarse level. Consequently if implementing such an approach, one would observe sensitivity of the method when δ changes.

Overall the revised multilevel method [7] for solving (6) is the following:

Algorithm 1 *Given z and an initial guess $\widetilde{u} = z$, with $L+1$ levels,*

1. *Iteration starts with $u_{old} = \widetilde{u}$ ($\widetilde{u}$ contains the initial guess before the first iteration and the updated solution at all later iterations).*

2. *Smooth for ν iterations the approximation on the finest level 1, i.e. solve (7) for $i, j = 1, 2, \ldots, n$.*

3. *Iterate for ν times on each coarse level $k = 2, 3, \ldots, L+1$:*

   - *compute $\widetilde{z} = z - \widetilde{u}$, $\widetilde{w}_{i,j}$, $\widetilde{v}_{m,\ell}$, and $\widetilde{h}_{m,\ell}$ via (12),*
   - *compute the minimizer c of (11) if $k \leq L$; or on the coarsest level $k = L+1$, the correction constant is simply $c = mean(\widetilde{w}) = mean(z - \widetilde{u})$,*
   - *add the correction, $\widetilde{u} = \widetilde{u} + P_k c$, where P_k is the interpolation operator distributing $c_{i,j}$ to the corresponding $b \times b$ block on level k as illustrated in (10).*

4. *On level $k = 1$, check the possible patch size for each position (i, j):*

$$patch = \{(i_\ell, j_\ell) \ : \ |u_{i_\ell, j_\ell} - u_{i,j}| < \varepsilon\}$$

   for some small ε. Implement the piecewise constant update as with Step 3.

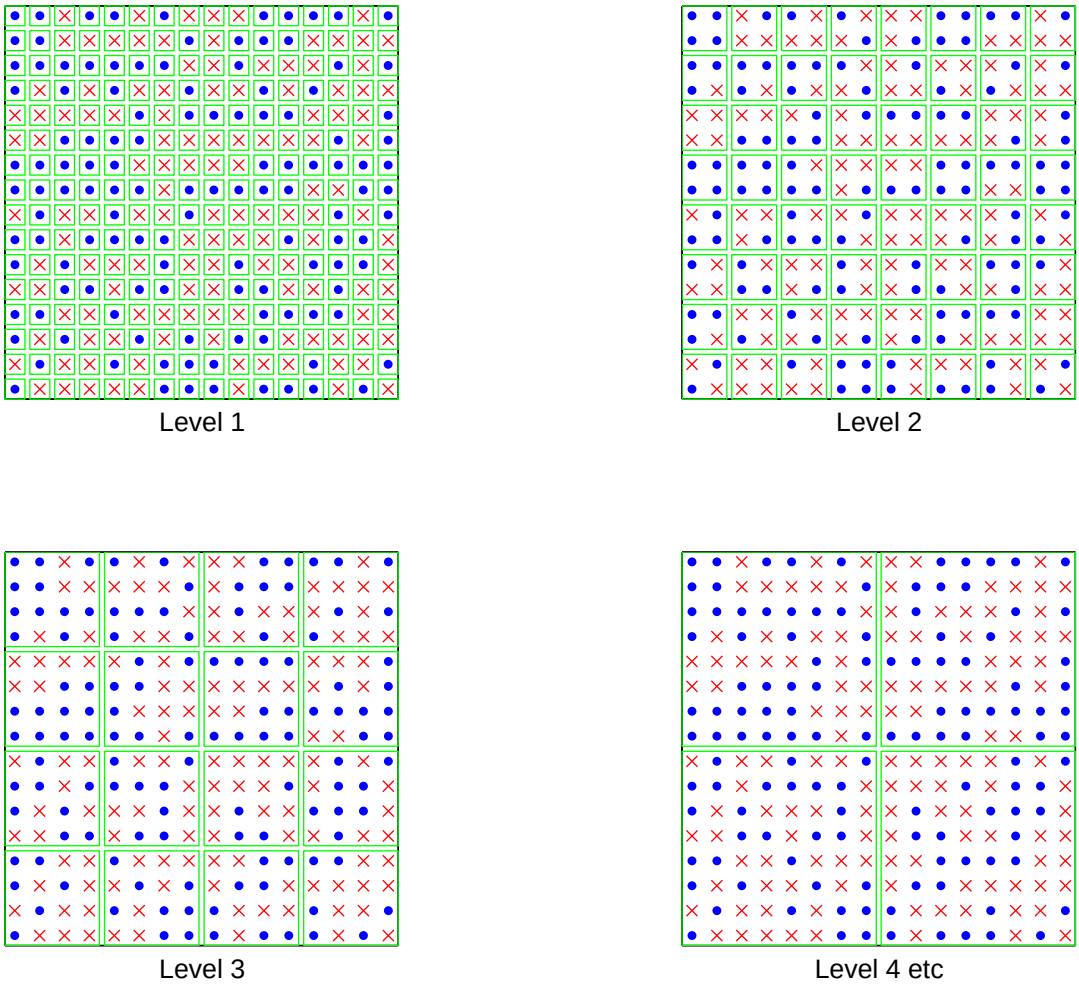
5. *If $\|\widetilde{u} - u_{old}\|_2$ is small enough, exit with $u = \widetilde{u}$ or return to Step 1 and continue iterations.*

We note that whenever the TV semi-norm is used (resulting in a non-smooth J), the solution will allow local constants. Such local constants lead to the hemi-variateness of the solution, which may prevent local minimizations reaching the global minimizer [17]. Step 4 here is to overcome this, see [8]. Finally we remark that the above method can also be adapted for solving the Poisson denoising model [3].

3 A modified multilevel method for the 2-phase approach

The main difference between our new problem (1) and problem (6) is that in (1) we only do the minimization in a subset $\mathcal{N}$, and we do not update any pixels in $\mathcal{N}^c$. Therefore we have to adapt the above multilevel method accordingly. Figure 1 shows a typical example of noisy pixels in $\mathcal{N}$ (denoted by $\times$) mixed with correct pixels in $\mathcal{N}^c$ (denoted by $\bullet$). Clearly some subproblems are

Figure 1: Illustration of subproblems for a piecewise constant multigrid method for impulse denoising.



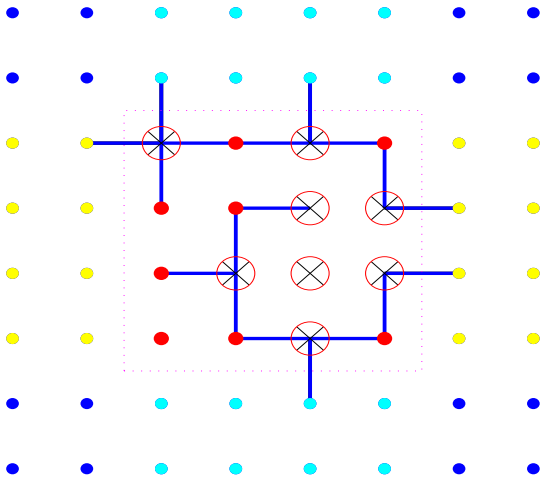
empty if all pixels within are the correct pixels. This will be the main characteristic of our new multilevel method.

First of all, we show how to use the coordinate descent method on the finest level 1

$$\begin{cases} \text{Given} & u^{(0)} = (u_{i,j}^{(0)}) = (z_{i,j}) \text{ with } l = 0, \\ \text{Solve} & \hat{c} = \operatorname{argmin}_{c\in\mathbb{R}} F^{\text{loc}}(u_{i,j}^{(l)} + c), \;\; u_{i,j}^{(l)} = u_{i,j}^{(l)} + \hat{c} \text{ for } (i,j) \in \mathcal{N} \\ \text{Set} & u^{(l+1)} = (u_{i,j}^{(l)}) \text{ and repeat the above step with } l = l+1 \\ & \text{until a prescribed stopping step on } l, \end{cases} \tag{13}$$

which modifies the method (9) for the related Gaussian denoising case, where minimization is

Figure 2: Illustration of interacting pixels in block $(2,2)$ in Figure 1 (level 3).



carried out for all pixels. Here the local merit functional is defined as

$$F^{\text{loc}}(u_{i,j}) = \sum_{(m,r)\in\mathcal{V}_{i,j}\cap\mathcal{N}^c} 2\phi(u_{i,j} - z_{m,r}) + \sum_{(m,r)\in\mathcal{V}_{i,j}\cap\mathcal{N}} \phi(u_{i,j} - u_{m,r}),$$

which is already in discrete form for $\phi(t)$ defined in (2)–(4). For TV-norm, it is defined as J^{loc} in (8), but we only do the minimization for $(i,j) \in \mathcal{N}$. The discretization and minimization will proceed as with J^{loc} before, with the only modification of not involving terms which are not linked to noisy pixels.

Next we consider how to formulate the coordinate descent method on level $k > 1$ (as in Figure 1). To introduce the main idea and formulation, we need to generalize the previous notation $\mathcal{V}_{i,j}$ from a single pixel to a block of pixels. Denote by $D(i,j)$ the index set of all pixels of block (i,j) on level k $(k \geq 1)$ and $\mathcal{V}_{D(i,j)}$ the set of neighbouring pixels of $D(i,j)$. Let $D(i,j) = B(i,j) \bigcup I(i,j)$ be a non-overlapping splitting, separating the boundary pixels $B(i,j)$ and the interior pixels $I(i,j)$ of block (i,j). Therefore on levels $k = 1, 2$, $I(i,j)$ is empty and $D(i,j) = B(i,j)$ for all (i,j). On level 1, clearly, $D(i,j) = \{(i,j)\}$ so $\mathcal{V}_{i,j} = \mathcal{V}_{D(i,j)}$ in this case. For an example on level 3, consider block $(i,j) = (2,2)$ as depicted in Figure 1: $D(2,2), B(2,2), I(2,2)$ contain 16, 12, 4 respective members and $\mathcal{V}_{D(2,2)}$ contains 20 members (neighbors).

To simplify the block minimization problem $\min_{c_{i,j}\in\mathbb{R}} F(\widetilde{u} + P_k c_{i,j})$ for block (i,j) on level k, we point out these characteristics:

1. only noisy pixels $D(i,j) \cap \mathcal{N}$ require taking into consideration in the minimization.

2. all noisy pixels in $B(i,j) \cap \mathcal{N}$ enter the minimization formula.

3. the noisy pixels in $I(i,j) \cap \mathcal{N}$ enter the minimization formula only if one of their neighbors is a correct pixel.

In Figure 2, we illustrate the above observations using the example of block $(2,2)$ from Figure 1. There are only 8 noisy pixels (in $D(2,2) \cap \mathcal{N}$ denoted by $\otimes$) within this block of 16 pixels and in particular all 5 members in $B(2,2) \cap \mathcal{N}$ will enter in the minimization formula while only 2 members (out of the total 3) in $I(2,2) \cap \mathcal{N}$ will enter in the minimization formula. This is because any two neighbouring interior pixels have the same difference in gray values before and after adding a constant, see the (3,3)th member in this block.

We are ready to state a simple and computable form of the block minimization problem $\min_{c_{i,j} \in \mathbb{R}} F(\widetilde{u} + P_k c_{i,j})$ for block (i,j). This problem is equivalent to minimizing the following

$$
\begin{aligned}
G^{\mathrm{loc}}(c_{i,j}) \;=\; & \sum_{(i_1,j_1) \in I(i,j) \cap \mathcal{N}} \; \sum_{(m,r) \in \mathcal{V}_{i_1,j_1} \cap \mathcal{N}^c} 2\phi(\widetilde{u}_{i,j} + c_{i,j} - z_{m,r}) \; + \\
& \sum_{(i_1,j_1) \in B(i,j) \cap \mathcal{N}} \left(\sum_{(m,r) \in \mathcal{V}_{i_1,j_1} \cap \mathcal{N}^c} 2\phi(\widetilde{u}_{i,j} + c_{i,j} - z_{m,r}) \; + \right. \\
& \left. \sum_{(m,r) \in \mathcal{V}_{i_1,j_1} \cap \mathcal{N} \cap \mathcal{V}_{D(i,j)}} \phi(\widetilde{u}_{i,j} + c_{i,j} - \widetilde{u}_{m,r}) \right) \\
= \; & \sum_{(i_1,j_1) \in D(i,j) \cap \mathcal{N}} \; \sum_{(m,r) \in \mathcal{V}_{i_1,j_1} \cap \mathcal{N}^c} 2\phi(\widetilde{u}_{i,j} + c_{i,j} - z_{m,r}) \; + \\
& \sum_{(i_1,j_1) \in B(i,j) \cap \mathcal{N}} \; \sum_{(m,r) \in \mathcal{V}_{i_1,j_1} \cap \mathcal{N} \cap \mathcal{V}_{D(i,j)}} \phi(\widetilde{u}_{i,j} + c_{i,j} - \widetilde{u}_{m,r}) \\
= \; & \sum_{(i_1,j_1) \in D(i,j) \cap \mathcal{N}} \; \sum_{(m,r) \in \mathcal{V}_{i_1,j_1} \cap \mathcal{N}^c} 2\phi(c_{i,j} - \widetilde{z}_{m,r}) \; + \\
& \sum_{(i_1,j_1) \in B(i,j) \cap \mathcal{N}} \; \sum_{(m,r) \in \mathcal{V}_{i_1,j_1} \cap \mathcal{N} \cap \mathcal{V}_{D(i,j)}} \phi(c_{i,j} - \widetilde{z}_{m,r}) \qquad (14)
\end{aligned}
$$

where $\widetilde{z}_{m,r} = z_{m,r} - \widetilde{u}_{i,j}$ for $(m,r) \in \mathcal{N}^c$ and $\widetilde{z}_{m,r} = \widetilde{u}_{m,r} - \widetilde{u}_{i,j}$ for $(m,r) \in \mathcal{N}$. Here (14) is a one-dimensional minimization problem for $c_{i,j}$ that may be solved by any available solver (as remarked in the previous section). Once solved, we add the constant correction: $\widetilde{u} = \widetilde{u} + P_k c_{i,j}$.

As in the Gaussian noise case, whenever the TV semi-norm is used, the solution will allow local constants, which may prevent local minimizations reaching the global minimizer [8, 17].

Following [8] (see Step 4 of Algorithm 1), we detect if such 'patch' exists:

$$H = \text{'patch'} = \Big\{(i_\ell, j_\ell) \;\Big|\; |\widetilde{u}_{i_\ell,j_\ell} - \widetilde{u}_{i,j}| < \varepsilon \;\text{ and }\; (i,j),(i_\ell,j_\ell) \in \mathcal{N}\Big\}. \tag{15}$$

for small ε (e.g. 10^{-3}). If this happens, the coordinate descent method can get stuck. Our solution is to let each patch of such pixels form a local block minimization problem. Assume the above patch H is embedded in some rectangular block $D(i,j)$ of pixel indices. The local block minimization problem is to find the best constant $c_{i,j}$ to be added to all the noisy candidates in $D(i,j)$ so that the overall merit functional is minimized. This will proceed exactly as in (14).

Our overall algorithm will proceed as follows:

Algorithm 2 *Given the observed image z, an initial guess $\widetilde{u}$, and the noisy candidate set $\mathcal{N}$.*

1. *Iteration starts with $u_{old} = \widetilde{u}$.*
2. *Smooth for ν iterations the approximation on the finest level* 1*, i.e. solve (13) for $i, j = 1, 2, \ldots, n$.*
3. *Iterate for ν iterations on each coarse level $k = 2, 3, \ldots, L+1$:*
   — *compute $\widetilde{z} = z - \widetilde{u}$*
   — *compute the minimizer c of (14)*
   — *add the correction, $\widetilde{u} = \widetilde{u} + P_k c$.*
4. *On level $k = 1$, find each patch H via (15), and implement the piecewise constant update as with Step* 3*.*
5. *If $\|\widetilde{u} - u_{old}\|_2$ is small enough, exit with $u = \widetilde{u}$ or return to Step* 1 *and continue iterations.*

In our experiments, we take $\nu = 2$.

To estimate the complexity of Algorithm 2, we need to estimate the number of terms in (14). For the given image $z \in \mathbb{R}^{n\times n}$, let the predicted noise level be $100w\%$ from phase 1; e.g. $w = 0.5$ for 50% noise. Firstly the cardinality of $\mathcal{N}$ is approximately wN with $N = n^2$ for an $n \times n$ image. Consequently the cardinality of $\mathcal{N}^c$ is $(1-w)N$. Secondly on level k, the number of operations associated all interior pixels $\mathcal{V}_{i_1,j_1} \cap \mathcal{N}^c$ will be $2wN$ while the number of terms associated all boundary pixels $\mathcal{V}_{i_1,j_1} \cap \mathcal{N} \cap \mathcal{V}_{D(i,j)}$ will be $4\frac{N}{b^2}(1-w)b = 4(1-w)N/2^{k-1}$. Therefore similarly

to [8] we can estimate the complexity of one cycle of Algorithm 2 as follows

$$sw\sum_{k=1}^{L+1}\left(2wN+4(1-w)N/2^{k-1}\right)\approx 2sN(L+1)w^2+8Nsw(1-w)\approx N\log(N),$$

which is close to the optimal $O(N)$ expected from a multilevel method. Here s is the total number of local relaxation for solving each local minimization per cycle (i.e. the total number from all ν iterations per cycle).

4 Numerical experiments

Here we shall demonstrate the usefulness of the proposed multilevel method (Algorithm 2). Restoration performance is quantitatively measured by the peak signal-to-noise ratio (PSNR)

$$PSNR = PSNR(r,u) = 10\log_{10}\frac{255^2}{\frac{1}{mn}\sum_{i,j}(r_{i,j}-u_{i,j})^2}$$

where $r_{i,j}$ and $u_{i,j}$ denote the pixel values of the original image and the restored image respectively, with $r, u \in \mathbb{R}^{m\times n}$. Here we assume $z_{i,j}, r_{i,j}, u_{i,j} \in [0, 255]$.

We will consider salt-and-pepper noise here and the noisy candidate set $\mathcal{N}$ is detected by the adaptive median filter (AMF), see [4, 13]. We note that AMF uses median values to restore pixels in $\mathcal{N}$ to obtain the restored image on $\mathcal{A}$. But for the 2-phase method, AMF is used just to obtain $\mathcal{N}$, and pixels in $\mathcal{N}$ are then restored by minimizing (1) in phase 2. We will consider three different ways of restoration here:

- MG — our Algorithm 2 with the initial image u on $\mathcal{A}$ given by AMF.
- NT — the Newton continuation method [5] with the initial image u on $\mathcal{A}$ given by AMF.
- MG/NT — Algorithm 2 with initial image u on $\mathcal{A}$ given by the solution obtained by NT.

We have three sets of experiments. In set 1, we compare the performance of MG/NT with NT for $\phi(t) = \sqrt{t^2+10^2}$ as used in [1]. In set 2, we compare the performance of MG with NT for the same ϕ. Finally in set 3, we consider the performance of MG with $\phi(t) = |t|$ and with the TV model (5). Since these functionals are non-smooth, such cases cannot be handled by NT [5] or conjugate gradient methods [1], unless we regularize the functional. It turns out that models (2) and (5) may produce better restored images than the popular choice (3) in cases of high noise level.

Table 1: Comparison of restoration quality of MG/NT with NT (δ: improvement)

Problem	Noise	PSNR(r, z)	PSNR(r, u_{AMF})	PSNR(r, u_{NT})	PSNR$(r, u_{MG/NT})$	δ
1	50%	8.51	32.41	38.95	39.49	0.53
	70%	7.05	28.17	33.66	35.09	1.42
	95%	5.73	19.66	23.32	25.70	2.38
2	50%	8.37	28.92	32.77	32.95	0.18
	70%	6.90	26.05	29.78	30.13	0.35
	95%	5.58	19.66	23.51	25.10	1.58
3	50%	7.79	27.86	31.27	31.15	−0.11
	70%	6.32	25.29	29.02	29.09	0.07
	95%	5.01	18.87	23.61	25.01	1.41
4	50%	8.47	33.76	40.65	40.79	0.13
	70%	7.02	29.55	35.41	36.76	1.35
	95%	5.69	19.97	24.21	27.24	3.03

Set 1 — **comparison of MG/NT with NT**. We have taken 4 test images of size 512×512 and experimented with various levels of salt-and-pepper type noise. We summarise the results in Table 1, where one can observe that the improvement by MG/NT over NT alone is more for higher noise levels. The comparative results for the noise level of 70% are shown in Figures 3 and 4. We note that the improvement can be bigger for smaller size images; we show in Figure 5 a test example with 64×64 resolution and 95% noise for an image taken from Problem 4 where the improvement in PSNR is about 6dB.

Set 2 — **comparison of MG with NT**. The previous tests show that our MG works best if it uses the result from NT as an initial guess. Here, we test how MG performs without such good initial guesses. We display the results in Table 2 where one can see that improvements are still observed in most cases, especially when the noise level is high. Clearly in most test cases, MG/NT gives the best restoration result.

In separate experiments, we have compared the speed of our MG with NT as 2 different solvers. It turns out that the CPU times are comparable for images up to the size of 512×512. However for larger images with $n \geq 1024$, NT cannot be run on the same Pentium PC with 2GB memory (due to excessive memory requirement) while MG can be run showing the same quality restoration after $2 - 3$ cycles.

Set 3 — **behaviour of MG/NT with non-smooth regularization functionals**. We now discuss how Algorithm 2 performs with models (2) and (5). We note that local minimization

Figure 3: Comparison of MG/NT with NT with 70% salt-and-pepper noise (problems 1 and 2).

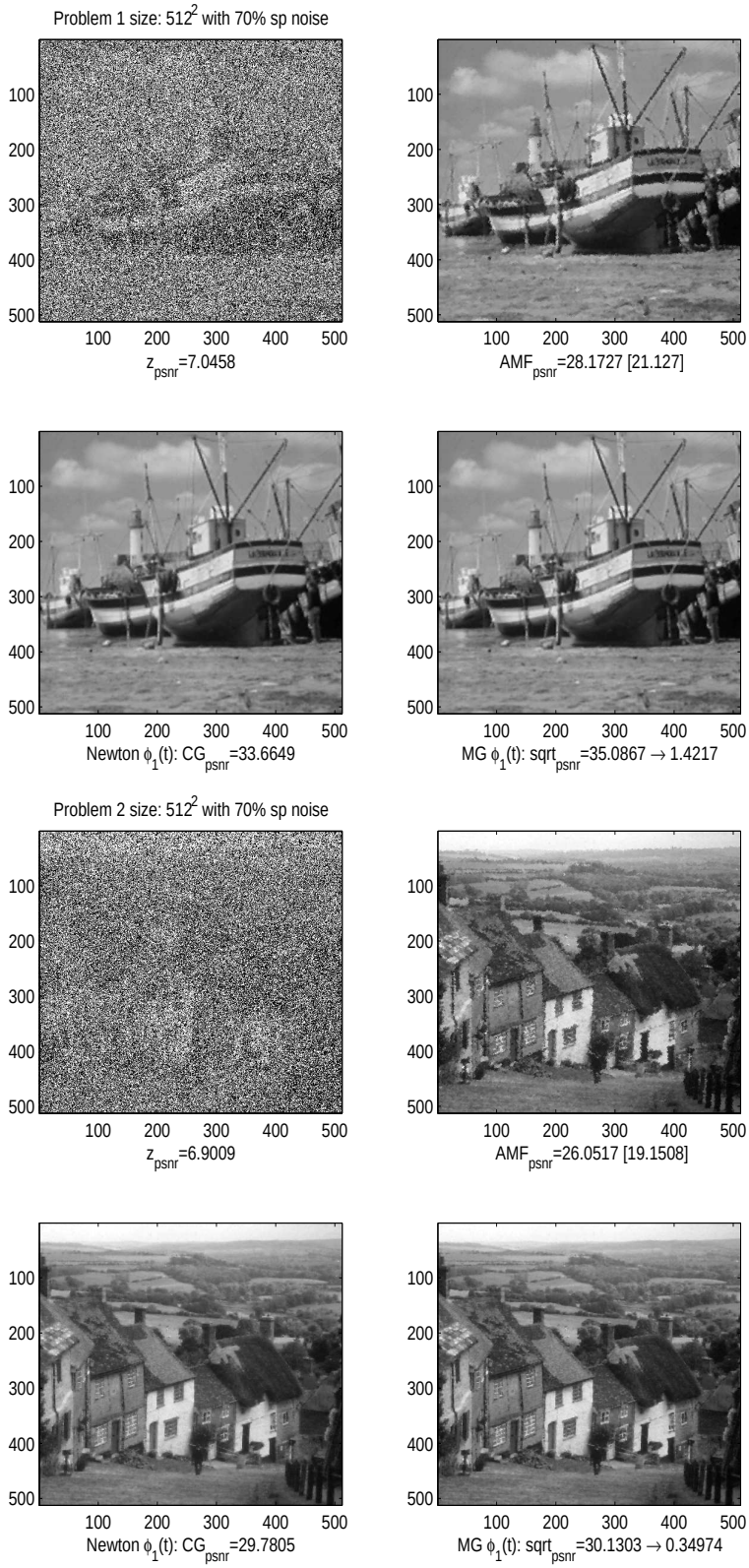


Figure 4: Comparison of MG/NT with NT with 70% salt-and-pepper noise (problems 3 and 4).

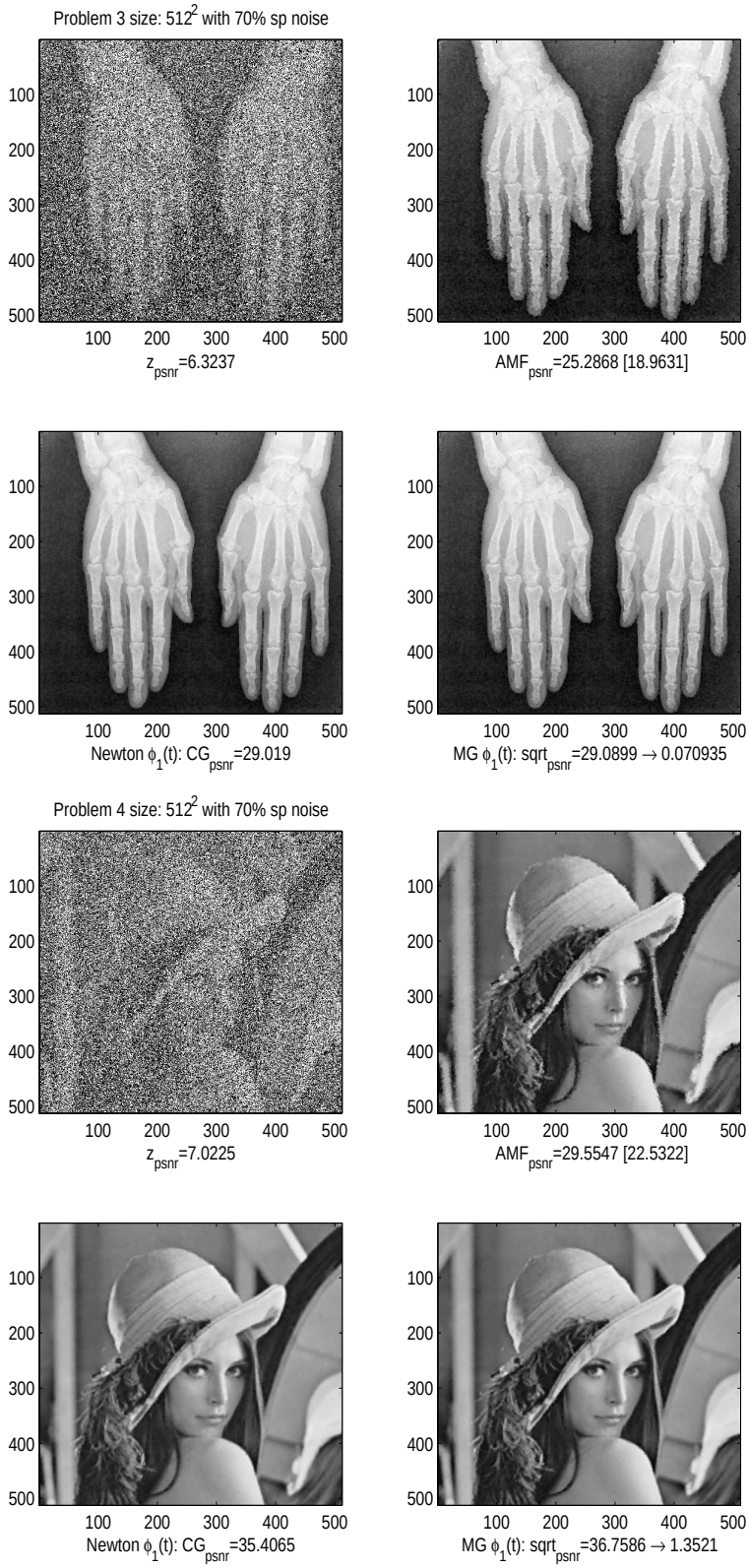


Figure 5: Comparison of MG/NT with NT with 95% for a small image.

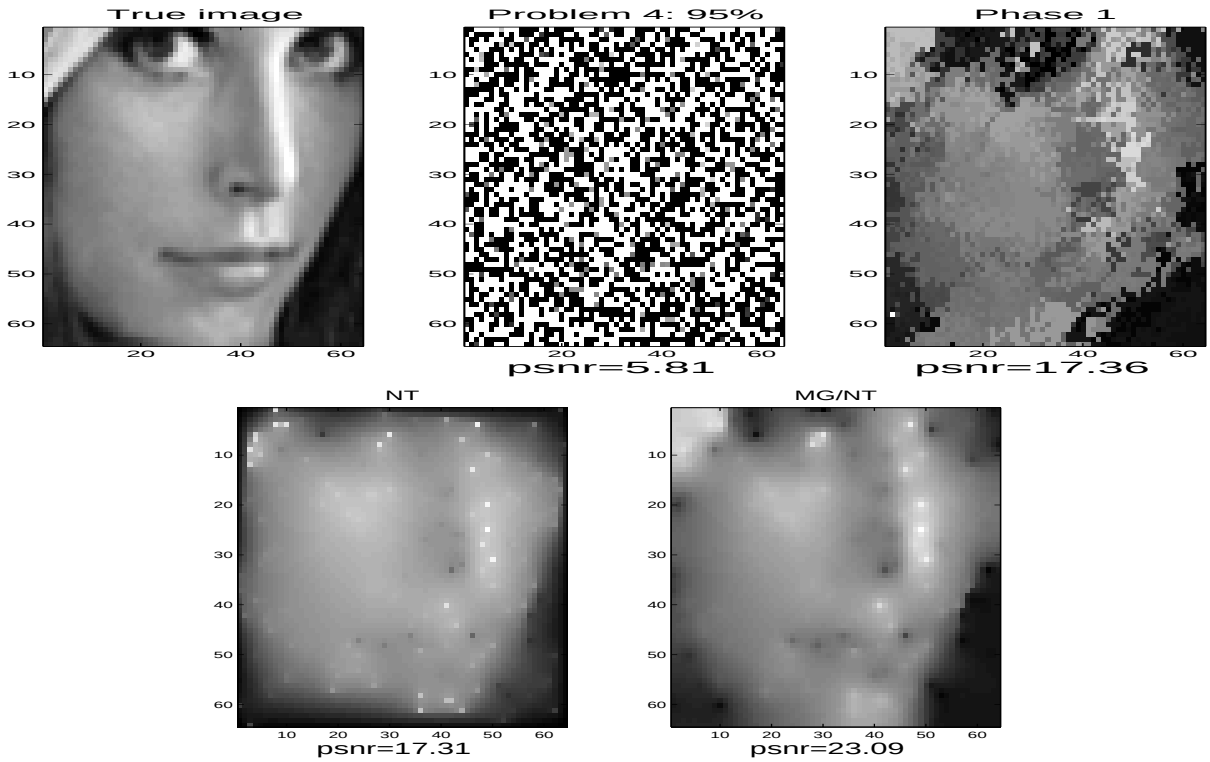


Table 2: Comparison of restoration quality of MG with NT (δ: improvement)

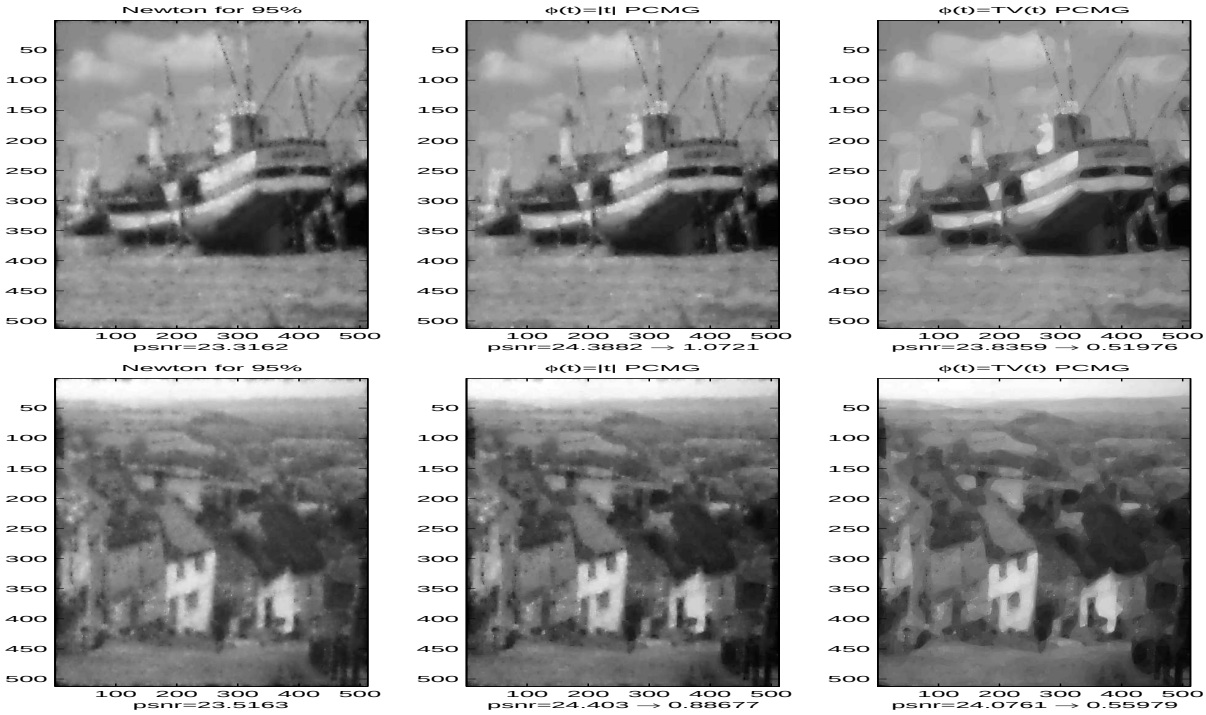
Problem	Noise	PSNR(r, u_{NT})	PSNR(r, u_{MG})	δ
1	50%	38.95	39.17	0.22
	70%	33.66	35.06	1.40
	95%	23.32	25.37	2.05
2	50%	32.77	32.42	−0.36
	70%	29.78	30.06	0.28
	95%	23.51	24.78	1.27
3	50%	31.27	30.63	−0.63
	70%	29.02	29.03	0.01
	95%	23.61	24.72	1.12
4	50%	40.65	40.53	−0.12
	70%	35.41	36.73	1.32
	95%	24.21	26.80	2.59

using model (2) leads to problems of the type

$$\min_{c\in\mathbb{R}} G(c), \qquad\qquad G(c) = \sum_{j=1}^{p} |c - c_j|$$

for some integer p, which has the exact solution $c = c_{\min} = \text{median}([c_1, \ldots, c_p])$. Our experiments show that MG/NT is still quite useful for high noise cases, as seen from Figure 6 for images with 95% noise, where we compare NT (the left plot), MG/NT with (2) (the middle plot) and model (5) (the right plot). We can observe that the improvement over NT is not as great as MG/NT with (3) in set 1 tests. However it is pleasing to see that such (previously not tested) non-smooth functionals can lead to better restored images.

Figure 6: Performance of MG/NT with models (2) and (5) with 95% salt-and-pepper noise. For problem 1: $\text{psnr}(r, u_{NT}) = 23.32$, $\text{psnr}(r, u_{eq.(2)}) = 24.39$ and $\text{psnr}(r, u_{TV}) = 23.86$. For problem 2: $\text{psnr}(r, u_{NT}) = 23.52$, $\text{psnr}(r, u_{eq.(2)}) = 24.40$ and $\text{psnr}(r, u_{TV}) = 24.08$.



Finally, we discuss on the number of levels that can be used in Algorithm 2. In previous tests, we used the full levels. This point arises as all local minimizations are to help achieve the global convergence in a less nested way than the geometric MG for a partial differential equation. As discussed before, we always need the patch level and the corresponding coarse level minimization on each patch block. We wish to see if other coarse levels are needed. In Figures 7–8, we display

respectively the cases of $\nu = 1$ and $\nu = 2$ for testing problems $1-2$ with 90% noise and $n = 512$. There, we compared the obtained PSNR values as the number of MG cycles increases when $levels = 1, 2$ and 10 (the maximal number of levels), plotting against the corresponding PSNR value from NT. Clearly one observes that there is no harm in using more MG levels, larger ν makes the use of full MG levels less necessary and above all convergence does not require to have all coarse levels. The related idea of starting iterations from the coarsest level rather than the finest is also tested (on the above 4 examples) and no advantages are observed. However for extremely simple images that contain a few large and piecewise constant features, it is not difficult to see that this (full multigrid-like) idea will be useful. In any case, we emphasize again that the patch level is always needed for convergence.

Figure 7: Convergence history for $\nu = 1$ with Problems 1 (left) and 2 (right).

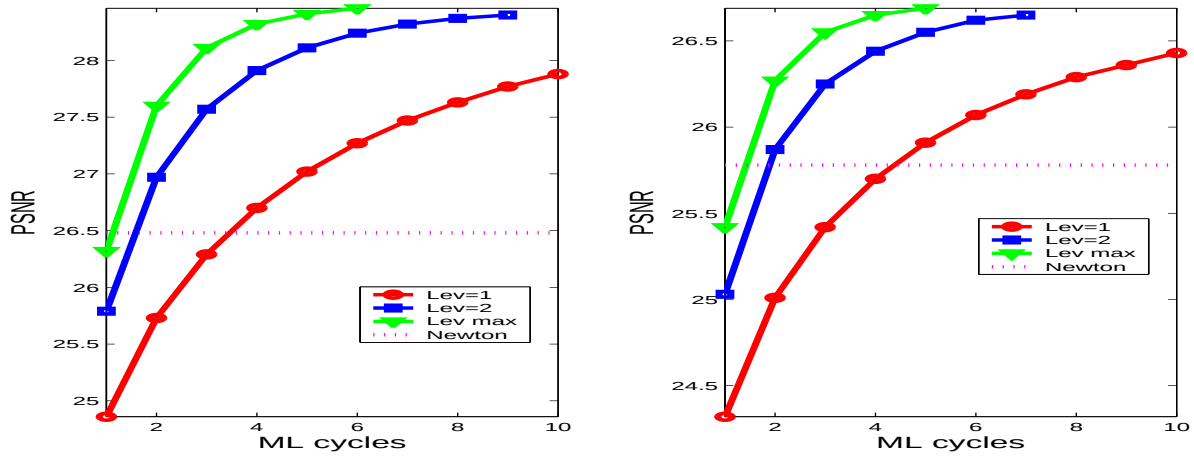
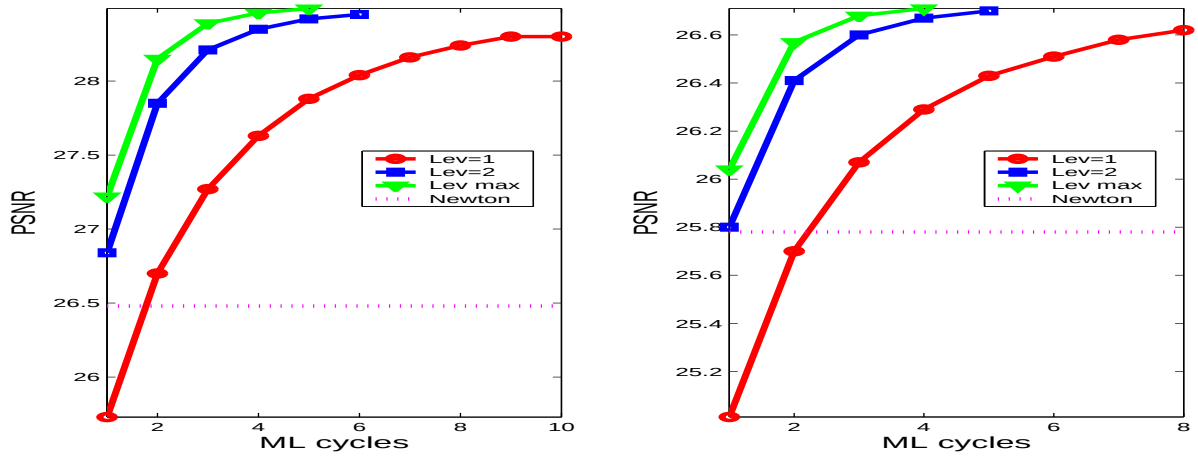


Figure 8: Convergence history for $\nu = 2$ with Problems 1 (left) and 2 (right).



Remark 1 *It is of interest to remark on why the initial image u at $\mathcal{N}$ should influence the convergence*

of Algorithm 2. We believe that the initial image u supplied by phase 1 obtained by local median type ideas commonly has over-estimated piecewise constant patches present in u that our existing multilevel method (Algorithm 2) cannot handle. Similar problem also existed for the piecewise constant multilevel method (Algorithm 1) for Gaussian denoising. For instance, if the true image (in 1 dimension) is $r = [10\ 10\ 10\ 10\ 20\ 20\ 20\ 20]$ *and an initial image is given as* $u = [10\ 10\ 10\ 10\ 10\ 10\ 20\ 20]$*, Algorithm 1 will not work while the initial image* $u = [1\ 1\ 1\ 1\ 50\ 50\ 50\ 50]$ *or any random vector u will be fine. This is related to the assumption of constant patches being correctly detected [8]. In this sense, one idea would be to ensure that the initial image does not have any constant patches at all.*

5 Conclusions

We have generalized a multilevel method previously proposed for the standard Gaussian denoising model to solve an impulse denoising model. The multi-resolution strategy of the new algorithm is found to give better restoration results for images with high noise level and the improvement is up to 2–3 dB. As the multilevel method has a nearly optimal complexity, it naturally offers a fast solution procedure for extremely large images.

Acknowledgements

The authors wish to thank J. F. Cai, CUHK, for providing various assistance in numerical experiments. They are grateful to all anonymous referees for making critical and helpful comments. The second author acknowledges the support from the Leverhulme Trust RF/9/RFG/2005/0482 as well as the support and hospitality of the Department of Mathematics, CUHK.

References

[1] J. Cai, R. Chan, and B. Morini (2006), *Minimization of edge-preserving regularization functional by conjugate gradient type methods*, Proc. 1st Int. Conf. PDE-based Image Proc., eds. X. C. Tai et al, Springer-Verlag, pp. 109–122.

[2] R. Chan, T. Chan and J. Wan (1997), *Multigrid for differential-convolution problems arising from image processing*, Proc. Workshop on Sci. Comput., eds: G. Golub, S. Lui, F. Luk, and R. Plemmons, Springer-Verlag, pp. 58–72.

[3] R. Chan and K. Chen (2007), *Multilevel algorithm for a Poisson noise removal model with total-variation regularisation*, to appear in: Int. J. Comp. Math.

[4] R. Chan, C. Ho, and M. Nikolova (2005), *Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization*, IEEE Trans. Image Proc., 14, 1479–1485.

[5] R. Chan, C. Ho, C. Leung, and M. Nikolova (2005), *Minimization of detail-preserving regularization functional by Newton's method with continuation*, Proc. IEEE Int. Conf. on Image Proc., Genova, Italy, pp. 125–128.

[6] R. Chan, C. Hu, and M. Nikolova (2004), *An iterative procedure for removing random-valued impulse noise*, IEEE Signal Proc. Letters, 11, 921–924.

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

[8] T. Chan and K. Chen (2006), *An optimization-based multilevel algorithm for total variation image denoising*, SIAM J. Multiscale Modeling and Simulations, 5, 615–645.

[9] T. Chan and J. Shen (2005), *Image Processing and Analysis—Variational, PDE, wavelet, and stochastic methods*, SIAM Publications, Philadelphia, USA.

[10] T. Chen and H. Wu (2001), *Space variant median filters for the restoration of impulse noise corrupted images*, IEEE Trans. Circuits and Systems II, 48, 784–789.

[11] M. Donatelli and S. Capizzano (2006), *On the regularizing power of multigrid-type algorithms*, SIAM J. Sci. Comput., 26, 2053–2076.

[12] Y. Dong, R. Chan, and S. Xu (2007), *A detection statistic for random-valued impulse noise*, IEEE Trans. Image Proc., 16, 1112–1120.

[13] H. Hwang and R. A. Haddad (1995), *Adaptive median filters: new algorithms and results*, IEEE Trans. Image Proc., 4, 499–502.

[14] M. Nikolova (2004), *A variational approach to remove outliers and impulse noise*, J. Math. Imaging and Vision, 20, 99–120.

[15] T. Nodes and N. Gallagher Jr. (1984), *The output distribution of median type filters*, IEEE Trans. Communications, COM-32.

[16] S. Osher and S. Esedoglu (2003), *Decomposition of images by the anisotropic Rudin-Osher-Fatemi model*, CAM report 03-34, UCLA, USA.

[17] J. Ortega and W. Rheinboldt (1970), *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, NY, USA.

[18] G. Pok, J. Liu, and A. Nair (2003), *Selective removal of impulse noise based on homogeneity level information*, IEEE Trans. Image Proc., 12, 85–92.

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

[20] C. Vogel (2002), *Computational Methods For Inverse Problems*, SIAM publications, USA.