International Journal of Computer Mathematics (2014)
Vol. 91, No. 3, 661–687, http://dx.doi.org/10.1080/00207160.2013.807340

Taylor & Francis
Taylor & Francis Group

A restarted iterative homotopy analysis method for two nonlinear models from image processing

Behzad Ghanbari[a], Lavdie Rada[b]* and Ke Chen[b]

[a]*Department of Mathematics, Kermanshah University of Technology, Kermanshah, Iran;* [b]*Centre for Mathematical Imaging Techniques, Department of Mathematical Sciences, University of Liverpool, Peach Street, Liverpool L69 7ZL, UK*

(*Received 20 November 2012; revised version received 5 March 2013; second revision received 3 May 2013; accepted 14 May 2013*)

Total variation (TV) minimization-based nonlinear models have been proven to be very useful and successful in image processing. A lot of effort has been devoted to overcome the nonlinearity of the model and at the same time to obtain fast numerical schemes. In this paper, we propose a restarted iterative homotopy analysis method (HAM) to improve the computational efficiency for the TV models and will show by experiments that this method demonstrates great potential for recovering the noise and with great speed in both image denoising and image segmentation models. The method modifies the existing HAM and makes it suitable to potentially solve other nonlinear partial differential equations arising from image processing models. In our examples, we will demonstrate the validity of a restarted HAM and that this method is efficient and robust even for images with large ratios of noise and with much less CPU time than other methods.

Keywords: image processing; total variation; denoising; image segmentation; discrete homotopy analysis method; finite difference scheme; series solution

2010 AMS Subject Classifications: 62H35; 65N22; 68U10; 35A15; 65C20; 74G65; 74G75; 35C10

1. Introduction

Digital images in various modalities are increasingly generated and used in practical applications, and presence of noise in them is inevitable. Denoising is a first pre-processing step in analysing images and corrects the image by removing the noise. A number of image processing techniques are proposed for removing image noise and preserving important image information such as edges [45]. The additive Gaussian noise is the first and most common problem in image processing. In other cases, uniformly distributed noise (or other types such as Poisson noise [11] and multiplicative noise [16]) may also appear. Denoising via linear filters normally does not perform satisfactorily, since both noise and edges contain high frequencies, while the nonlinear denoising models have been applied and found to be successful. Rudin and Osher [37] and Rudin *et al.* [38] first introduced the total variation (TV) norm as an edge preserving model. This model

*Corresponding author. Email: ladirada@liv.ac.uk, lavdie.rada@liv.ac.uk

is a successful tool for image restoration (including both denoising and deblurring) and at the same time for edge enhancement (segmentation) too. High-order effective models [6,54] improving TV have been proposed, but their nonlinear partial differential equations (PDEs) bring some implementation and speed difficulties [51].

The Euler–Lagrange equation associated with the TV functional for those models is a nonlinear PDE. Due to the difficulties (or even impossibility) in obtaining analytic solutions of nonlinear PDEs, with strong nonlinearity in particular, it is easier to get a numerical approximation of a given nonlinear problem. In the last few years, there has been much progress in developing new implicit or explicit algorithms for nonlinear models with their own positive and negative feedback. In cases when an implicit method has been applied as the solver, we may face the following problems: (i) convergence is dependent on some properties of the matrix of the system after digitization; (ii) implementation is complicated for general nonlinear equations; and (iii) large storage is required. On the other hand, an explicit method (such as the easily implemented time-marching (TM) model) requires the time step to be chosen sufficiently small to get convergence and speeding up would be a challenge.

A relatively new method, namely homotopy analysis method (HAM), first introduced in 1992 [22], attracts with a simple way of controlling and adjusting the convergence region and rate of solution series of nonlinear problems. This method has been improved [23–25,27–29] and widely used to solve linear or nonlinear ordinary differential equation (ODE) and simple PDE problems showing great performance. Different fields in science, engineering, technological fields and finance have employed the method to solve or improve many types of problem [1,2,17,18,20,39,52] or even to find some new solutions of a few nonlinear equations which have never been solved by previous analytic methods or even numerical methods [26]. In comparison with well-known methods such as the perturbation techniques method [34,35] or the so-called non-perturbation techniques, such as the δ-expansion method [4], Adomian's decomposition method [3] and so on, HAM does not depend on small/large physical parameters and is valid not only for weakly nonlinear problems, but also for strongly nonlinear problems, and at the same time, the convergence of the solution has been proved [27]. Unfortunately, applying it to realistic nonlinear PDE models does not suit due to the analytical difficulties in working out the high-order approximations demanded by HAM. Sorting this out and at the same time using HAM's quality was the motivation of our work.

In this paper, we develop a restarted HAM (RHAM) and show that it has the same efficiency, or even better, compared with HAM, while being simple and easy to apply. The method avoids the requirement of working with high-order terms as in HAM, by proposing to use the sequence of three linear equations and then restart the HAM process. We will demonstrate through examples that RHAM is compatible with HAM for solving hard inverse problems (such as image processing) and nonlinear PDEs. The paper aims to answer these questions: (i) does RHAM perform well for general nonlinear equations; (ii) does it mathematically bring a stable scheme; (iii) is the quality of the model as good, or even better, compared with other methods; (iv) can it be a general form and (v) can it deal with large images, where speed is crucial? In this paper, we present the application of RHAM for the evolution approaches of TV-based models, which suggest that it may be used for other unexplored hard nonlinear applications.

In what follows, in Section 2, we first review the general idea of the HAM. Then, the classical TV restoration model and its discrete form are presented in Section 3. In this section, we will develop and present the restarted iterative HAM for the model. Part of this section will be answering simple questions such as 'How RHAM deals with simple cases?'. In Section 4, we describe the segmentation model before we apply the developed discrete HAM to it. In Section 5, we present various numerical results obtained from the implementation of RHAM for both denoising restoration and segmentation models, and then comparisons are made between obtained results and those of TM and the additive operator splitting (AOS) method.

2. Review of an HAM

In this section, we briefly introduce the standard HAM for a general nonlinear problem. This will be the first step to apply the RHAM (as a discrete form of HAM). This method has been known as an analytic method for solving nonlinear problems [27].

Consider a nonlinear equation of the following form:

$$\mathcal{N}[u(\mathbf{x},t)] = 0 \tag{1}$$

subject to the initial condition

$$u(\mathbf{x},0) = u_0(\mathbf{x}), \tag{2}$$

where $\mathcal{N}$ is a nonlinear operator which represents the whole equation, $\mathbf{x} \in R^n$, t denotes independent variable and $u(\mathbf{x})$ is an unknown function. Based on [24], the following zero-order deformation equation is constructed from the original equation (1) as follows:

$$(1-q)\mathcal{L}[\varphi(\mathbf{x},t;q) - u_0(\mathbf{x},t)] = q\hbar\mathcal{H}(\mathbf{x},t)\mathcal{N}[\varphi(\mathbf{x},t;q)], \tag{3}$$

where $u_0(\mathbf{x},t)$ is an initial guess, $\hbar$ is an auxiliary parameter, $q \in [0,1]$ is an embedding parameter, $\varphi(\mathbf{x},t;q)$ is a function of t and q, $\mathcal{H}$ is a non-zero auxiliary function and $\mathcal{L}$ is an auxiliary linear operator with the following property:

$$\mathcal{L}[\varphi(\mathbf{x},t)] = 0 \quad \text{when} \quad \varphi(\mathbf{x},t) = 0. \tag{4}$$

It should be emphasized that we have the freedom to choose the initial approximation, the auxiliary linear operator $\mathcal{L}$, the auxiliary parameter $\hbar$ and the auxiliary function $\mathcal{H}$. Obviously, since $\hbar \neq 0$, $\mathcal{H} \neq 0$, when $q = 0$ and $q = 1$, it holds that $\varphi(\mathbf{x};0) = u_0(\mathbf{x})$ and $\varphi(\mathbf{x};1) = u(\mathbf{x})$, respectively. Thus, as q increases from 0 to 1, the solution $\varphi(\mathbf{x};q)$ deforms from the initial guess $u_0(\mathbf{x})$ to the solution $u(\mathbf{x})$. Expanding $\varphi(\mathbf{x};q)$ in the Taylor series with respect to q, one has

$$\varphi(\mathbf{x},t;q) = u_0(\mathbf{x},t) + \sum_{m=1}^{\infty} u_m(\mathbf{x},t)q^m, \tag{5}$$

where

$$u_m(\mathbf{x},t) = \frac{1}{m!}\frac{\partial\varphi^m(\mathbf{x},t;q)}{\partial q^m}\bigg|_{q=0}. \tag{6}$$

If the auxiliary linear operator, the initial guess, the auxiliary parameter $\hbar$ and the auxiliary function $\mathcal{H} \neq 0$ are chosen such that the series (5) converges at $q = 1$, one has

$$u(\mathbf{x},t) = u_0(\mathbf{x},t) + \sum_{m=1}^{\infty} u_m(\mathbf{x},t), \tag{7}$$

which must be one of the solutions of the original nonlinear equation, as proved by Liao [22].

Let us define $\vec{u}_k(\mathbf{x}) = \{u_0(\mathbf{x}), \ldots, u_k(\mathbf{x})\}$ as the vector of the composed solutions. According to the definition for $u_m(\mathbf{x},t)$ (Equation (6)), the governing equation and the corresponding initial condition of $u_m(\mathbf{x},t)$ can be deduced from the zero-order deformation equation (3) and Equation (4) as follows. Differentiating the zero-order deformation equation (3) m-times with respect to q, dividing by $m!$ and finally setting $q = 0$, we obtain the following mth-order deformation problem:

$$\mathcal{L}[u_m(\mathbf{x},t) - \chi_m u_{m-1}(\mathbf{x},t)] = \hbar\mathcal{H}R_m[\vec{u}_{m-1}(\mathbf{x},t)], \tag{8}$$

where

$$R_m[\vec{u}_{m-1}(\mathbf{x},t)] = \frac{1}{(m-1)!}\frac{\partial^{m-1}\mathcal{N}[\varphi(\mathbf{x},t;q)]}{\partial q^{m-1}}\bigg|_{q=0} \tag{9}$$

and

$$\chi_m = \begin{cases} 1, & m > 1, \\ 0, & m \leqslant 1. \end{cases}$$

Applying the inverse operator $\mathcal{L}^{-1}$ on both sides of Equation (8), we can obtain

$$u_m(\mathbf{x},t) = \chi_m u_{m-1}(\mathbf{x},t) + \mathcal{L}^{-1}\{\hbar\mathcal{H}R_m[\vec{u}_{m-1}(\mathbf{x},t)]\} \tag{10}$$

and at the Mth order we have $u_M(\mathbf{x},t) = \sum_{m=0}^{M} u_m(\mathbf{x},t)$. Apparently as M increases we may obtain a more accurate approximate solution of the original equation (1). Note that when solving the mth-order deformation equations (8), $R_m[\vec{u}_{m-1}(\mathbf{x},t)]$ depends only on $u_0, u_1, u_2, \ldots, u_{m-1}$ which are already known.

In our applications, we will use two different linear operators as follows[1]:

$$\mathcal{L}_1(\varphi(\mathbf{x},t;q)) = \frac{\partial\varphi(\mathbf{x},t;q)}{\partial t} + \theta\varphi(\mathbf{x},t;q) \tag{11}$$

and

$$\mathcal{L}_2(\varphi(\mathbf{x},t;q)) = (1+t)\frac{\partial\varphi(\mathbf{x},t;q)}{\partial t} + \varphi(\mathbf{x},t;q) \tag{12}$$

with the property $\mathcal{L}_1[C_1\mathrm{e}^{-t}] = 0$ and $\mathcal{L}_2[C_2/(1+t)] = 0$, and θ is a positive constant. More analysis can be done to optimize the parameter θ, introduced in the term $\mathcal{L}_1$, and to give answer to the experimental result for different θ but this is not the main concern of this manuscript.

Below we shall write u^{HAM1} and u^{HAM2}, notation used to make the distinction between the linear operators $\mathcal{L}_1$ and $\mathcal{L}_2$, respectively, to find the solution $u(\mathbf{x}) = u(x,y)$ of Equation (1).

By applying the inverse operators $\mathcal{L}_1^{-1}$ and $\mathcal{L}_2^{-1}$ to both sides of the high-order deformation equation (8), subject to the initial condition

$$u_m(x,y,0) = 0,$$

the mth terms of the solution are, respectively, obtained in the following forms:

$$u_m^{\mathrm{HAM1}}(x,y,t) = \chi_m u_{m-1}^{\mathrm{HAM1}}(x,y,t) + \hbar\exp(-\theta t)\int_0^t \mathcal{H}_1(\tau)R_m[\vec{u}_{m-1}^{\mathrm{HAM1}}(x,y,t,\tau)]\,\mathrm{d}\tau \tag{13}$$

and

$$u_m^{\mathrm{HAM2}}(x,y,t) = \chi_m u_{m-1}^{\mathrm{HAM2}}(x,y,t) + \frac{\hbar}{1+t}\int_0^t \mathcal{H}_2(\tau)R_m[\vec{u}_{m-1}^{\mathrm{HAM2}}(x,y,t,\tau)]\,\mathrm{d}\tau, \tag{14}$$

where $R_m[\vec{u}_{m-1}]$ is as defined by Equation (9). Indeed, the solution $u(x,y)$ of the original nonlinear equation (1), while using the linear operators $\mathcal{L}_1(u)$ and $\mathcal{L}_2(u)$, respectively, is expressed in the following form:

$$u^{\mathrm{HAM1}}(x,y,t) = \sum_{m=0}^{+\infty} a_m\exp(-m\theta t) \tag{15}$$

and

$$u^{\mathrm{HAM2}}(x,y,t) = \sum_{m=0}^{+\infty}\frac{b_m}{(1+t)^m}, \tag{16}$$

where the a_m, b_m's are coefficients depending on x and y.

According to the rule of solution expression denoted by Equation (15) and from Equation (13), the auxiliary function $H(\tau)$ should be in the form $H(\tau) = \mathrm{e}^{-k\tau}$, where k is an integer. It is found that, when $k \leqslant 1$, the solution of the high-order deformation equation (13) contains the term $t\mathrm{e}^{-t}$, which incidentally disobeys the rule of solution expression (15). When $k > 2$, the base e^{-2t} always disappears in the solution expression of the high-order deformation equation (13), so that the coefficient of the term e^{-2t} cannot be modified even if the order of approximation tends to infinity. Therefore, according to the so-called rule of coefficient ergodicity by Liao [24], we have to set $k = 2$, which uniquely determines the corresponding auxiliary function $\mathcal{H}_1(\tau) = \exp(-2\tau)$.

Following the same discussion above for the rule of solution expression denoted by Equation (16) and from Equation (14), we get $\mathcal{H}_2(\tau) = 1/(1+\tau)^2$.

Thus, starting from the initial approximation $u_0(x, y, t) = u_0(x, y)$, we can use the recurrence formulas (13) and (14) to successively obtain $u_m(t)$ for $m \geqslant 1$.

The HAM has been applied all these years to continuous functions because of the freedom to use different base functions to approximate a nonlinear problem, its validity for nonlinear problems, and its convenient way to adjust the convergence region and the rate of the approximation series. The first attempt at a discrete version of the HAM was made recently by Zhu *et al.* [55] for a diffusion equation.

We find the method attractive and at the same time challenging to apply in image processing techniques, such as denoising and image segmentation, which brings us to nonlinear PDEs which have some difficulties. The question we had to deal with at this point was: 'Is it easy to carry on with $m = 3$, $m = 4, \ldots, m = M$?', for nonlinear PDEs arising from this kind of technique. The derivation of the R_m term with respect to q does not bring a higher order than the given PDE, but might require not only calculation but even some difficulties in programming. To avoid all this: 'What if we stop at a certain $m (m = 2$ in our case) and restart with a better approximation?' In this way for $M = m = 2$ a restarting scheme with $R = 5$ has the form $\hat{u_M} = u_0^{(1)} + \sum_{r=1}^{R} \sum_{m=1}^{M} u_m^{(r)}$. This is the simple idea of the so-called RHAM which will be given in detail in the following sections for both image denoising and image segmentation.

3. A restarted iterative HAM for total variational denoising

The presence of noise in images is unavoidable and for this reason, a good denoising technique is required. In practice, the classical algorithms based on least squares, Fourier series or the $\mathcal{L}^2$ norm approximation do not approximate images containing edges well. To preserve image edges and features, a TV denoising model was introduced by Rudin *et al.* [38], which is based on a variational problem using the TV norm as a nonlinear non-differentiable functional and a TM scheme to solve the Euler–Lagrange equation arising from it. This model preserves the edges very well but the problem is nonlinear and there are computational difficulties which make the method slow. Because of edge preserving qualities the method immediately became successful and in this way attracted attention. The algorithms for solving this nonlinear problem can be split in two main kinds: *Parabolic type*, which is easy to be programmed and as a fastest algorithm, the AOS method can be used in this group, and *Elliptic type*, which includes a fixed-point iteration technique proposed by Vogel and Oman [48], a primal–dual method by Chan *et al.* [14] or a nonlinear multigrid method introduced by Savage and Chen [40,41] and Vogel [47] (and references therein). Other models improved the TV model [38], by different regularizers [10,32], or extended to higher orders [7,30,31,44]. One issue in these models is to allow a small β in Equation (20) when solving the nonlinear equation. There is some progress in this direction [51]. The main result of this work concerns an iterative procedure designed to reduce the complexity of the TV model [38] for the parabolic algorithm and to achieve a faster speed of convergence than AOS. We

will show that this method has the potential to be used for different nonlinear differential equations and for different orders. Our interest is in the fast iterative procedure of the inverse problem arising from the TV model [38], by using an RHAM which brings the solution of nonlinear PDEs to the solution of a linear system and the result of the algorithm shows great performance. Our idea is to replace the variational problem by a sequence of approximations decomposing it into the solution of a linear equation. We will test our model using two criteria: the quality of the restoration results and the speed of the iterative solution.

3.1 *The TV model*

Let Ω be a bounded domain with intensity function $z(x, y)$ (known), denoting the pixel values of an image, contaminated with an additive noise $\eta(x, y)$ (unknown).[2] We would like to construct $u(x, y)$ (unknown), the desired clean image for the observed image $z(x, y)$, such that

$$z(x, y) = u(x, y) + \eta(x, y). \tag{17}$$

The objective of denoising algorithms is to reconstruct $u(x, y)$ from $z(x, y)$. Image denoising is an ill-conditioned inverse problem and the denoising methods use regularization techniques to approximate $u(x, y)$. In the constrained minimization problem, it is well known that the L_1 norm of the gradient is the appropriate space. This is basically the space of functions of bounded total variation. In the given problem, we need to minimize the noise.

Rudin *et al*. [38] proposed the use of the following minimization problem:

$$\min_u \int_\Omega |\nabla u(x, y)| \, \mathrm{d}x \, \mathrm{d}y + \frac{\lambda}{2} \int_\Omega (u(x, y) - z(x, y))^2 \, \mathrm{d}x \, \mathrm{d}y, \tag{18}$$

where $\lambda > 0$ is a tuning parameter. The first term is the TV of $u(x, y)$, a regularizer, while the second term is a fidelity term ensuring that the denoised image $u(x, y)$ will be close to the given image $z(x, y)$. The fitting parameter λ is important for balancing denoising and smoothing; therefore, it depends on the noise level. As a straightforward observation, we can say that, in the presence of high noise, the parameter λ has to be decreased. Large λ corresponds to very little noise removal and small λ yields a blurry, over-smoothed restoration $u(x, y)$. Equation (18) leads to the Euler–Lagrange equation, a nonlinear elliptic PDE with homogeneous Neumann boundary conditions as follows:

$$\begin{aligned} g(u(x, y)) = \nabla \cdot \left(\frac{\nabla u(x, y)}{|\nabla u(x, y)|} \right) - \lambda (u(x, y) - z(x, y)) = 0, \quad (x, y) \in \Omega, \\ \nabla u(x, y) \cdot \vec{n} = 0, \quad (x, y) \in \partial\Omega, \end{aligned} \tag{19}$$

where $\vec{n}$ is the unit normal vector exterior to the boundary $\partial\Omega$.

To avoid division by zero in numerical implementation, we replace the non-differentiable term $|\nabla u(x, y)|$ by a smooth approximation $|\nabla u(x, y)|_\beta = \sqrt{|\nabla u(x, y)|^2 + \beta^2}$ for some small $\beta > 0$. Therefore, we get

$$\begin{aligned} g(u(x, y)) = \nabla \cdot \left(\frac{\nabla u(x, y)}{|\nabla u(x, y)|_\beta} \right) - \lambda (u(x, y) - z(x, y)) = 0, \quad (x, y) \in \Omega, \\ \nabla u(x, y) \cdot \vec{n} = 0, \quad (x, y) \in \partial\Omega. \end{aligned} \tag{20}$$

3.2 *A TM (gradient descent) method*

For solving the Euler–Lagrange equation (20), Rudin *et al*. [38] used a parabolic equation as a solution procedure, which means we solve

$$\frac{\partial u(x,y;t)}{\partial t} = \nabla \cdot \left(\frac{\nabla u(x,y;t)}{|\nabla u(x,y;t)|_\beta} \right) - \lambda(u(x,y;t) - z(x,y)), \quad t > 0, \ \ (x,y) \in \partial\Omega,$$
$$\nabla u \cdot \vec{n} = 0, \quad (x,y) \in \partial\Omega, \quad \text{where } u(x,y,0) \text{ is given.} \tag{21}$$

For numerical implementation, let us assume that the domain Ω has been split into $N \times M$ cells where the grid points are located at $(x_i = ih_x, y_j = jh_y)$, $i = 1 \ldots N, j = 1 \ldots M$, $t_k = k\Delta t$, where Δt and $k = 1, 2, \ldots$ denote the time step and iteration time, respectively. We denote the values of $u(x,y,t)$ at the grid points (x_i, y_j, t_k) by u_{ij}^k, and $u_{ij}^0 = z(x_i, y_j)$.

For simplicity and without loss of generality, we assume that $M = N$, $h_x = h_y = 1$. In this way, the curvature term can be approximated by

$$\kappa(u_{i,j}^k) = \nabla \cdot \left(\frac{\nabla u}{|\nabla u|_\beta} \right)\Bigg|_{i,j} = \left(\frac{\partial}{\partial x}\left[\frac{u_x}{|\nabla u_\beta|} \right] + \frac{\partial}{\partial y}\left[\frac{u_x}{|\nabla u_\beta|} \right] \right)\Bigg|_{i,j}$$
$$= \left[\Delta_-^x \left(\frac{\Delta_+^x u_{ij}^k}{\sqrt{(\Delta_+^x u_{ij}^k)^2 + (\Delta_+^y u_{ij}^k)^2 + \beta}} \right) + \Delta_-^y \left(\frac{\Delta_+^y u_{ij}^k}{\sqrt{(\Delta_+^y u_{ij}^k)^2 + (\Delta_+^y u_{ij}^k)^2 + \beta}} \right) \right] \tag{22}$$

with

$$\Delta_\mp^x u_{ij} = \mp(u_{i\mp1j} - u_{ij}),$$
$$\Delta_\mp^y u_{ij} = \mp(u_{ij\mp1} - u_{ij})$$

and $g(u_{i,j}^k)$ calculated in the following form:

$$g(u_{i,j}^k) = \kappa(u_{i,j}^k) - \lambda(u_{ij}^k - z_{ij}). \tag{23}$$

If the time derivative u_t at $(i, j, k\Delta t)$ is approximated by the forward difference as

$$(u_t)_{ij}^k = \frac{u_{ij}^{k+1} - u_{ij}^k}{\Delta t},$$

then by considering Equation (23), we obtain

$$\frac{u_{ij}^{k+1} - u_{ij}^k}{\Delta t} = g(u_{i,j}^k) \tag{24}$$

or

$$u_{ij}^{k+1} = u_{ij}^k + \Delta t g(u_{i,j}^k) \tag{25}$$

with boundary condition $u_{0j}^k = u_{1j}^k$, $u_{Nj}^k = u_{N-1j}^k$, $u_{i0}^k = u_{i1}^k$ and $u_{iN}^k = u_{iN-1}^k$ for $i, j = 1, 2, \ldots, N$.

The TM method for solving Equation (21) is described in Algorithm 1:

Algorithm 1 The TM method: $u^k \leftarrow \mathrm{TM}(u^k, z, \lambda, \mathrm{maxit}, \mathrm{tol})$

for iter $= 1 :$ maxit **do**
Compute $g(u_{i,j}^k)$ by using Equation (23)
Perform TM steps on linear system by Equation (25)
$u_{i,j}^{k+1} \leftarrow u_{i,j}^k + \Delta t g(u_{i,j}^k)$
If $\|u^k - u^{k+1}\| < \mathrm{tol}$ or $\mathrm{PSNR}(u^k) > \mathrm{PSNR}(u^{k+1})$, set $u_{i,j}^k \leftarrow u_{i,j}^{k+1}$, Break;
$u_{i,j}^k \leftarrow u_{i,j}^{k+1}$.

end for

The problem with the TM approach is that, due to stability restrictions, the time step must be taken to be very small, resulting in very slow convergence. For this reason, we will compare our method with a fast solver too, such as the AOS method [49] which is an alternating direction implicit method [36].

3.3 *Difficulties with HAM*

The difficulty in applying the HAM for the ROF model comes from the nonlinear term

$$\nabla \cdot \left(\frac{\nabla u(x,y)}{|\nabla u(x,y)|_\beta} \right).$$

In real-life application dealing with this kind of term is quite normal. High-order PDEs can be even more complicated. For example, let us consider the curvature model

$$\alpha \nabla \cdot \left(\frac{\nabla \Phi'(\kappa)}{|\nabla u|_\beta} - \frac{\nabla u \cdot \nabla \Phi'(\kappa)}{(|\nabla u|_\beta)^3} \nabla u \right) + u - z = 0 \quad \text{in } \Omega \tag{26}$$

with κ the curvature of the image and Φ defined as $\Phi(\kappa) = |\kappa|$, $\Phi(\kappa) = \kappa^2$ or as in [6,53] as a combination of both. When applying the Mth-order HAM method, we have to differentiate M times the original nonlinear equation $\mathcal{N}(u)$ with respect to q. These derivatives, as we mentioned before, do not bring a higher order than the given PDE, but might require calculation as well as some more difficulties in programming by adding terms.

Stopping at a certain M (depending on hardness of the given nonlinear problem) and restarting with a better approximation will avoid this problem. In this way, we have a scheme of the form

$$\hat{u_M} = u_0^{(1)} + \sum_{r=1}^{R} \sum_{m=1}^{M} u_m^{(r)}$$

with restarting iteration number R.

Note: Before going to complicated image processing cases, an immediate question is 'Does RHAM work for simple cases such as ODEs?'. For this reason, we tested the idea for simpler problems, already taken in consideration with HAM and shown to be successful. In Appendix, RHAM shows great performance for these examples in comparison with HAM. The results show that a second-order RHAM performs as 8th- or 10th-order HAM.

3.4 *A RHAM for denoising*

Considering the above nonlinear equation (21) we define the nonlinear PDE as

$$\mathcal{N}[u(x,y,t)] = \frac{\partial u(x,y;t)}{\partial t} - \nabla \cdot \left(\frac{\nabla u(x,y;t)}{|\nabla u(x,y;t)|_\beta} \right) + \lambda(u(x,y;t) - z(x,y)) = 0. \tag{27}$$

The RHAM can be used to solve the equation by choosing the nonlinear operator $\mathcal{N}[\varphi(x,y,t;q)]$ in a straightforward manner

$$\mathcal{N}[\varphi(x,y,t;q)] = \frac{\partial \varphi(x,y,t;q)}{\partial t} - \nabla \cdot \left(\frac{\nabla \varphi(x,y,t;q)}{|\nabla \varphi(x,y,t;q)|_\beta} \right) + \lambda(\varphi(x,y,t;q) - z(x,y)), \tag{28}$$

where $0 \le q \le 1$.

In what follows, we will consider applying the two linear operators $\mathcal{L}_1$ and $\mathcal{L}_2$ introduced in Section 2 for solving the nonlinear PDE (21), and as initial approximation we choose

$$\varphi(x,y,t=0) = u_0(x,y) = z(x,y).$$

Since for large enough values of t, the solution of Equation (21) converges to the solution of Equation (19), we expect that the solution of Equation (21) has a finite value as t tends to infinity. In what follows, we give the solutions of Equations (13) and (14) for $m = 1$ and 2.

Keeping the notation u_0 for u_0^{RHAM1} and u_0^{RHAM2}, and u_1 for u_1^{RHAM1} and u_1^{RHAM2}, respectively, while applying Equation (9) in the following calculation of $R_1[\vec{u}]$ and $R_2[\vec{u}]$ and considering Equation (6) we have, respectively, for $m = 1$ and 2

$$R_1[\vec{u}_0] = \mathcal{N}(u_0) = -g(u_0), \tag{29}$$

$$\begin{aligned} R_2[u_1,u_0] &= \left.\frac{\partial \mathcal{N}[\varphi(\mathbf{x},t;q)]}{\partial q}\right|_{q=0} = \left.\frac{\partial}{\partial q}\left(\frac{\partial \varphi}{\partial t} - \nabla\cdot\left(\frac{\nabla\varphi}{|\nabla\varphi|_\beta}\right) + \lambda(\varphi - z)\right)\right|_{q=0} \\ &= \left.\left(\frac{\partial}{\partial t}\frac{\partial \varphi}{\partial q} - \frac{\partial}{\partial x}\left(\frac{\partial}{\partial q}\frac{\varphi_x}{|\nabla\varphi|_\beta}\right) - \frac{\partial}{\partial y}\left(\frac{\partial}{\partial q}\frac{\varphi_y}{|\nabla\varphi|_\beta}\right) + \frac{\partial(\lambda(\varphi - z))}{\partial q}\right)\right|_{q=0} \\ &= u_{1t} - \frac{\partial}{\partial x}\left(\frac{u_{1x}}{|\nabla u_0|_\beta} - \frac{u_{0x}u_{1x} + u_{0y}u_{1y}}{|\nabla u_0|_\beta^3}u_{0x}\right) \\ &\quad - \frac{\partial}{\partial y}\left(\frac{u_{1y}}{|\nabla u_0|_\beta} - \frac{u_{0x}u_{1x} + u_{0y}u_{1y}}{|\nabla u_0|_\beta^3}u_{0y}\right) + \lambda u_1. \end{aligned}$$

Similar to Equations (13) and (14), we get

for RHAM1

$$u_1^{\mathrm{RHAM1}} = \hbar \exp(-\theta t)\int_0^t \exp((\theta-2)\tau) R_1[\vec{u}_0]\,\mathrm{d}\tau = -\hbar\frac{\mathrm{e}^{-2t} - \mathrm{e}^{-\theta t}}{\theta - 2} g(u_0), \tag{30}$$

$$u_2^{\mathrm{RHAM1}} = u_1^{\mathrm{RHAM1}} + \hbar \exp(-\theta t)\int_0^t \exp((\theta-2)\tau) R_2[u_1^{\mathrm{RHAM1}}, u_0]\,\mathrm{d}\tau, \tag{31}$$

for RHAM2

$$u_1^{\mathrm{RHAM2}} = \frac{\hbar}{(1+t)}\int_0^t \frac{1}{(1+\tau)^2} R_1[\vec{u}_0]\,\mathrm{d}\tau = -\hbar\frac{t}{(1+t)^2} g(u_0), \tag{32}$$

$$u_2^{\mathrm{RHAM2}} = u_1^{\mathrm{RHAM2}} + \frac{\hbar}{1+t}\int_0^t \frac{1}{(1+\tau)^2} R_2[u_1^{\mathrm{RHAM2}}, u_0]\,\mathrm{d}\tau. \tag{33}$$

Both terms u_2^{RHAM1} and u_2^{RHAM2} can be easily found in the explicit form. In what follows, we give some more details about RHAM2 and in the same way RHAM1 can be calculated. The first term obtained, u_1^{RHAM2}, is a composition of two functions, one depending on the variable t and the other one on (x, y), so we can write $u_1^{\mathrm{RHAM2}} = -\hbar(t/(1+t)^2)g(u_0) = f(t)w(x, y)$, with $f(t) = -\hbar(t/(1+t)^2)$ and $w(x, y) = g(u_0)$. In this way, the $R_2[u_1, u_0]$ term can be rewritten as

$$\begin{aligned} R_2[u_1, u_0] &= f'(t)w - f(t)\frac{\partial}{\partial x}\left(\frac{w_x}{|\nabla u_0|_\beta} - \frac{u_{0x}w_x + u_{0y}w_y}{|\nabla u_0|_\beta^3}u_{0x}\right) \\ &\quad - f(t)\frac{\partial}{\partial y}\left(\frac{w_y}{|\nabla u_0|_\beta} - \frac{u_{0x}w_x + u_{0y}w_y}{|\nabla u_0|_\beta^3}u_{0y}\right) + \lambda f(t)w, \\ R_2[u_1, u_0] &= f'(t)w - f(t)\{S(w, w_x, w_y, u_0)\}, \end{aligned}$$

where

$$\begin{aligned} S(w, w_x, w_y, u_0) &= \frac{\partial}{\partial x}\left(\frac{w_x}{|\nabla u_0|_\beta} - \frac{u_{0x}w_x + u_{0y}w_y}{|\nabla u_0|_\beta^3}u_{0x}\right) \\ &\quad + \frac{\partial}{\partial y}\left(\frac{w_y}{|\nabla u_0|_\beta} - \frac{u_{0x}w_x + u_{0y}w_y}{|\nabla u_0|_\beta^3}u_{0y}\right) + \lambda w \end{aligned}$$

and from Equation (33), we have

$$u_2^{\mathrm{RHAM2}} = -\hbar\frac{t}{(1+t)^2}w + \frac{\hbar}{1+t}\left[w\hbar\frac{1-2t}{6(1+t)^4} - S(w, w_x, w_y, u_0)\hbar\frac{1+3t}{6(1+t)^3}\right].$$

An immediate observation is that the computational cost of the first-order approximation of the RHAM is as expensive as the TM. The second-order term of RHAM, u_2, brings the extra computation of the term $S(w, w_x, w_y, u_0)$ which is obtained explicitly[3] without having too much extra computation since the curvature term $w(x, y)$ has been already evaluated in the same step.

Easily we can note that the first-order approximation of the RHAM, that is, $m = 1$, coincide with the TM method. In this way, for $m = 1$, HAM and consequently RHAM have the same

Algorithm 2 RHAM solution expressed by exponential functions
RHAM1: $u^k \leftarrow \mathrm{RHAM1}(u^k, z, \lambda, \beta, \theta, \Delta t, \mathrm{maxit}, \mathrm{tol})$

Initialization: Given $u_{i,j}^0 = u_{i,j,0} = z_{i,j}$,
for $k = 1 : \mathrm{maxit}$ **do**
Set $u_{i,j,0}^k := u_{i,j}^{k-1}$
Compute $g(u_{i,j,0}^k)$ by using Equation (23)
Compute $u_{i,j,1}^k$ by using Equation (30)
Compute $u_{i,j,2}^k$ by using Equation (31)
$u_{i,j}^k \leftarrow u_{i,j,0}^k + u_{i,j,1}^k + u_{i,j,2}^k$
If $\|u^k - u^{k-1}\| < \mathrm{tol}$ or $\mathrm{PSNR}(u^{k-1}) > \mathrm{PSNR}(u^k)$, set $u_{i,j}^k \leftarrow u_{i,j}^{k-1}$, Break;
$u_{i,j}^k \leftarrow u_{i,j}^{k-1}$,

end for

stability condition. For $m > 1$, the stability analysis has been considered and detailed in the HAM framework and which can be used as a prediction in advance for the RHAM. More detailed analysis for RHAM will be considered in future.

The numerical implementation for RHAM1 from Equations (30) and (31) can be given in the form of an algorithm as described in Algorithm 2. In the same way, considering Equations (32) and (33) we get Algorithm 3 which describes RHAM2.

Algorithm 3 RHAM solution expressed by fractional functions
RHAM2: $u^k \leftarrow RHAM2(u^k, z, \lambda, \beta, \Delta t, maxit, tol)$

Initialization: Given $u^0_{i,j} = u_{i,j,0} = z_{i,j}$,
for $k = 1 :$ maxit **do**
Set $u^k_{i,j,0} := u^{k-1}_{i,j}$
Compute $g(u^k_{i,j,0})$ by using Equation (23)
Compute $u^k_{i,j,1}$ by using Equation (32)
Compute $u^k_{i,j2}$ by using Equation (33)
$u^k_{i,j} \leftarrow u^k_{i,j,0} + u^k_{i,j,1} + u^k_{i,j,2}$
If $\|u^k - u^{k-1}\| <$ tol or $\mathrm{PSNR}(u^{k-1}) > \mathrm{PSNR}(u^k)$, set $u^k_{i,j} \leftarrow u^{k-1}_{i,j}$, Break;
$u^k_{i,j} \leftarrow u^{k-1}_{i,j}$,

end for

As we see from the results in Section 5, RHAMs are effective for solving nonlinear equations such as Equation (28). There are many models using the TV operators as in Equation (28) which might be solved by RHAM. Below we give one such application of RHAM to a TV variational model in image segmentation.

4. Application of our RHAM to a variational image segmentation model

There has been a great attraction in recent years to the level set method for image segmentation. Object segmentation in images using variational methods was first introduced by Kass *et al*. [21], and its algorithm is known as the *snake algorithm.* This model was further developed as the geodesic active contours model and the level set method [8,9,42], while in contrast a nonlinear functional was introduced by Mumford–Shah [33] which was later implemented by Chan–Vese [12] and continuously improved by Chan and Vese [13], Chan *et al*. [15], Hintermuller and Ring [19], Tsai *et al*. [43] and Vese and Chan [46]. These latter methods consider the image as piecewise continuous functions that are surrounded by discontinuities represented as contours Γ and are known for better performance in the presence of noise than former methods.

The variational image segmentation model for the segmentation of the objects in an image implemented by Chan–Vese [12] considers the image as a piecewise constant

$$u(x, y) = \begin{cases} c_1 = \mathrm{mean}(\mathrm{inside}(\Gamma)) \\ c_2 = \mathrm{mean}(\mathrm{outside}(\Gamma)) \end{cases}$$

and the Mumford–Shah [33] model reduces to

$$\inf_{c_1,c_2,\Gamma} F(\Gamma, c_1, c_2) = \mu \,\text{length}(\Gamma) + \lambda_1 \int_{\text{inside}(\Gamma)} |z(x,y) - c_1|^2 \,\mathrm{d}x\,\mathrm{d}y + \lambda_2 \int_{\text{outside}(\Gamma)} |z(x,y) - c_2|^2 \,\mathrm{d}x\,\mathrm{d}y, \tag{34}$$

where $z(x, y)$ is the original image, c_1 and c_2 are the average values of z inside and outside of the contour Γ, and μ, λ_1 and λ_2 are non-negative fixed parameters.

In the following, we will deal with the Chan–Vese region-based model and show how to implement the RHAM.

Denoting the regularized Heaviside function and Delta function, respectively, as

$$H_\varepsilon(x) = \frac{1}{2}\left(1 + \frac{2}{\pi}\arctan\left(\frac{x}{\varepsilon}\right)\right), \quad H'_\varepsilon(x) = \delta_\varepsilon(x) = \frac{1}{\pi}\frac{\varepsilon}{\varepsilon^2 + x^2} \tag{35}$$

and representing Γ by the zero level set of the Lipschitz function $\phi(x, y) : \Omega \to \mathbb{R}$ as in [12], we get the Euler–Lagrange equation for $\phi(x, y)$ by the minimization of the energy $F(\Gamma, c_1, c_2)$ with respect to $\phi(x, y)$, c_1, c_2. The Euler–Lagrange equation is written as follows:

$$\begin{aligned} \delta_\epsilon(\phi(x,y))\left[\mu\nabla\cdot\left(\frac{\nabla\phi(x,y)}{|\nabla\phi(x,y)|}\right) - f_0(\phi(x,y))\right] = 0 \quad &\text{in } \Omega, \\ \frac{\delta_\epsilon(\phi(x,y))}{|\nabla\phi(x,y)|}\frac{\partial\phi(x,y)}{\partial\vec{n}} = 0 \quad &\text{on } \partial\Omega, \end{aligned} \tag{36}$$

where $\vec{n}$ is the unit normal exterior to the boundary $\partial\Omega$, $\partial\phi/\partial\vec{n}$ is the normal derivative of ϕ at the boundary and

$$f_0(\phi(x,y)) = \lambda_1(z(x,y) - c_1(\phi(x,y)))^2 - \lambda_2(z(x,y) - c_2(\phi(x,y)))^2 \tag{37}$$

with

$$c_1(\phi(x,y)) = \frac{\int_\Omega z(x,y)H_\epsilon(\phi(x,y))\,\mathrm{d}x\,\mathrm{d}y}{\int_\Omega H_\epsilon(\phi(x,y))\,\mathrm{d}x\,\mathrm{d}y} \tag{38}$$

$$c_2(\phi(x,y)) = \frac{\int_\Omega z(x,y)(1 - H_\epsilon(\phi(x,y)))\,\mathrm{d}x\,\mathrm{d}y}{\int_\Omega (1 - H_\epsilon(\phi(x,y)))\,\mathrm{d}x\,\mathrm{d}y} \tag{39}$$

(i.e the curve has a non-empty interior and exterior in Ω).

Note that the main equation (36) has the nonlinear TV operator present. In a similar way to Equation (28), we can consider to choose the nonlinear operator

$$\mathcal{N}[\varphi(x,y,t;q)] = \frac{\partial\varphi}{\partial t} - \delta_\epsilon(\varphi)\left[\mu\nabla\cdot\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right) - f_0(\varphi)\right] \tag{40}$$

with the initial approximation for the level set $\phi(x, y)$

$$\varphi(x,y,t=0) = \phi_0(x,y) = \sqrt{(x - x_0)^2 + (y - y_0)^2} - r_0.$$

The main difference is the change of notation from u_0, u_1, u_2 to ϕ_0, ϕ_1 ϕ_2. In the similar way to denoising, we apply the two linear operators (11) and (12) regarding the level set function ϕ. From Equation (9), we derive $R_1[\vec{\phi}_0]$ and $R_2[\vec{\phi}_0]$

$$R_1[\phi_0] = \mathcal{N}(\phi_0) = -\delta_\epsilon(\phi_0)\left[\mu\nabla\cdot\left(\frac{\nabla\phi_0}{|\nabla\phi_0|}\right) - f_0(\phi_0)\right]. \tag{41}$$

$$\begin{aligned}
R_2[\phi_0,\phi_1] &= \left.\frac{\partial\mathcal{N}[\varphi(x,y,t;q)]}{\partial q}\right|_{q=0} = \left.\frac{\partial}{\partial q}\left(\frac{\partial\varphi}{\partial t} - \delta_\epsilon(\varphi)\left[\mu\nabla\cdot\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right) - f_0(\varphi)\right]\right)\right|_{q=0}\\
&= \left.\left(\frac{\partial}{\partial q}\frac{\partial\varphi}{\partial t} - \frac{\partial}{\partial q}\left(\delta_\epsilon(\varphi)\right)\left[\mu\nabla\cdot\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right) - f_0(\varphi)\right]\right.\right.\\
&\quad \left.\left. -\delta_\epsilon(\varphi)\frac{\partial}{\partial q}\left[\mu\nabla\cdot\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right) - f_0(\varphi)\right]\right)\right|_{q=0}\\
&= \left.\left(\frac{\partial}{\partial t}\frac{\partial\varphi}{\partial q} - \frac{\partial\varphi}{\partial q}\delta'_\epsilon(\varphi)\left[\mu\nabla\cdot\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right) - f_0(\varphi)\right]\right.\right.\\
&\quad \left.\left. -\delta_\epsilon(\varphi)\left[\mu\frac{\partial}{\partial q}\nabla\cdot\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right) - \frac{\partial f_0(\varphi)}{\partial q}\right]\right)\right|_{q=0}\\
&= \phi_{1t} - \phi_1\delta'_\epsilon(\phi_0)\left[\mu\nabla\cdot\left(\frac{\nabla\phi_0}{|\nabla\phi_0|}\right) - f_0(\phi_0)\right]\\
&\quad -\delta_\epsilon(\phi_0)\left[\mu\frac{\partial}{\partial x}\left(\frac{\phi_{1x}}{|\nabla\phi_0|_\beta} - \frac{\phi_{0x}\phi_{1x}+\phi_{0y}\phi_{1y}}{|\nabla\phi_0|_\beta^3}\phi_{0x}\right)\right.\\
&\quad \left. +\mu\frac{\partial}{\partial y}\left(\frac{\phi_{1y}}{|\nabla\phi_0|_\beta} - \frac{\phi_{0x}\phi_{1x}+\phi_{0y}\phi_{1y}}{|\nabla\phi_0|_\beta^3}\phi_{0y}\right) - f'_0(\phi_0)\right],
\end{aligned}$$

where

$$\delta'_\epsilon(\phi_0) = \frac{\varepsilon}{\pi}\frac{-2\phi_0}{(\varepsilon^2+\phi_0^2)^2} \tag{42}$$

and[4]

$$f'_0(\phi_0) = -2c'_1(\phi_0)\lambda_1(z-c_1(\phi_0)) + 2c'_2(\phi_0)\lambda_2(z-c_2(\phi_0)). \tag{43}$$

To obtain $f'_0(\phi_0)$, we also need to calculate

$$c'_1(\phi_0) = \frac{(\int_\Omega z\delta_\epsilon(\phi_0)\phi_1\,\mathrm{d}x\,\mathrm{d}y)(\int_\Omega H_\epsilon(\phi_0)\,\mathrm{d}x\,\mathrm{d}y) - (\int_\Omega \delta_\epsilon(\phi_0)\phi_1\,\mathrm{d}x\,\mathrm{d}y)(\int_\Omega zH_\epsilon(\phi_0)\,\mathrm{d}x\,\mathrm{d}y)}{(\int_\Omega H_\epsilon(\phi_0)\,\mathrm{d}x\,\mathrm{d}y)^2}, \tag{44}$$

$$c'_2(\phi_0) = \frac{\begin{array}{c}-(\int_\Omega z\delta_\epsilon(\phi_0)\phi_1\,\mathrm{d}x\,\mathrm{d}y)(\int_\Omega(1-H_\epsilon(\phi_0))\,\mathrm{d}x\,\mathrm{d}y) + (\int_\Omega \delta_\epsilon(\phi_0)\phi_1\,\mathrm{d}x\,\mathrm{d}y)\\ (\int_\Omega z(1-H_\epsilon(\phi_0))\,\mathrm{d}x\,\mathrm{d}y)\end{array}}{(\int_\Omega(1-H_\epsilon(\phi_0))\,\mathrm{d}x\,\mathrm{d}y)^2}, \tag{45}$$

where $\delta_\varepsilon(\phi_0)$ is as defined in Equation (35).

By setting $m = 1$ in Equations (13) and (14), we, respectively, get *for RHAM1*

$$\phi_1^{\mathrm{RHAM1}} = \hbar \exp(-\theta t) \int_0^t \exp((\theta - 2)\tau) R_1[\vec{u}_0] \, \mathrm{d}\tau$$
$$= -\hbar \frac{\mathrm{e}^{-2t} - \mathrm{e}^{-\theta t}}{\theta - 2} \delta_\epsilon(\phi_0) \left[\mu \nabla \cdot \left(\frac{\nabla \phi_0}{|\nabla \phi_0|} \right) - f_0(\phi_0) \right], \tag{46}$$

$$\phi_2^{\mathrm{RHAM1}} = \phi_1^{\mathrm{RHAM1}} + \hbar \exp(-\theta t) \int_0^t \exp((\theta - 2)\tau) R_2[\phi_0, \phi_1^{\mathrm{RHAM1}}] \, \mathrm{d}\tau, \tag{47}$$

Figure 1. Test Set-1 Example 1. From the top row to the bottom, 10%, 15% and 20% random noise has been added to the image. For each row, the first image shows the result obtained with the TM method for the same number of iterations as the maximum iterations used for RHAM1 and RHAM2, the second image shows the result obtained with the TM method, and the third and the fourth images show the results obtained for RHAM1 and RHAM2.

for RHAM2

$$\phi_1^{\mathrm{RHAM2}} = \frac{\hbar}{(1+t)}\int_0^t \frac{1}{(1+\tau)^2} R_1[\vec{u}_0]\,\mathrm{d}\tau$$
$$= -\hbar\frac{t}{(1+t)^2}\delta_\epsilon(\phi_0)\left[\mu\nabla\cdot\left(\frac{\nabla\phi_0}{|\nabla\phi_0|}\right) - f_0(\phi_0)\right], \tag{48}$$

$$\phi_2^{\mathrm{RHAM2}} = \phi_1^{\mathrm{RHAM2}} + \frac{\hbar}{1+t}\int_0^t \frac{1}{(1+\tau)^2} R_2[\phi_0, \phi_1^{\mathrm{RHAM2}}]\,\mathrm{d}\tau, \tag{49}$$

which can be easily found in the explicit form.

The algorithms should thus be similar to Algorithms 2 and 3.

Figure 2. Test Set-1 Example 2. From the top row to the bottom, 10%, 15% and 20% random noise has been added to the image. For each row, the first image shows the result obtained with the TM method for the same number of iterations as the maximum iterations used for RHAM1 and RHAM2, the second image shows the result obtained with the TM method, and the third and the fourth images show the results obtained for RHAM1 and RHAM2.

5. Experimental results for the denoising and image segmentation problems using the RHAM method

In order to illustrate the speed and accuracy of the proposed denoising and segmentation algorithms, we provide experimental results with different images such as artificial, synthetic and real image. The comparison will be with TM [38] and AOS method [50]. To measure *the image quality*, the parameters *SNR* (the signal-to-noise ratio) and *PSNR* (the peak signal-to-noise ratio) can be used.

SNR of a noisy image is a measure of how much noise is presented in the image, and the *PSNR* measures how close the images are to each other. These quantities are given by the following

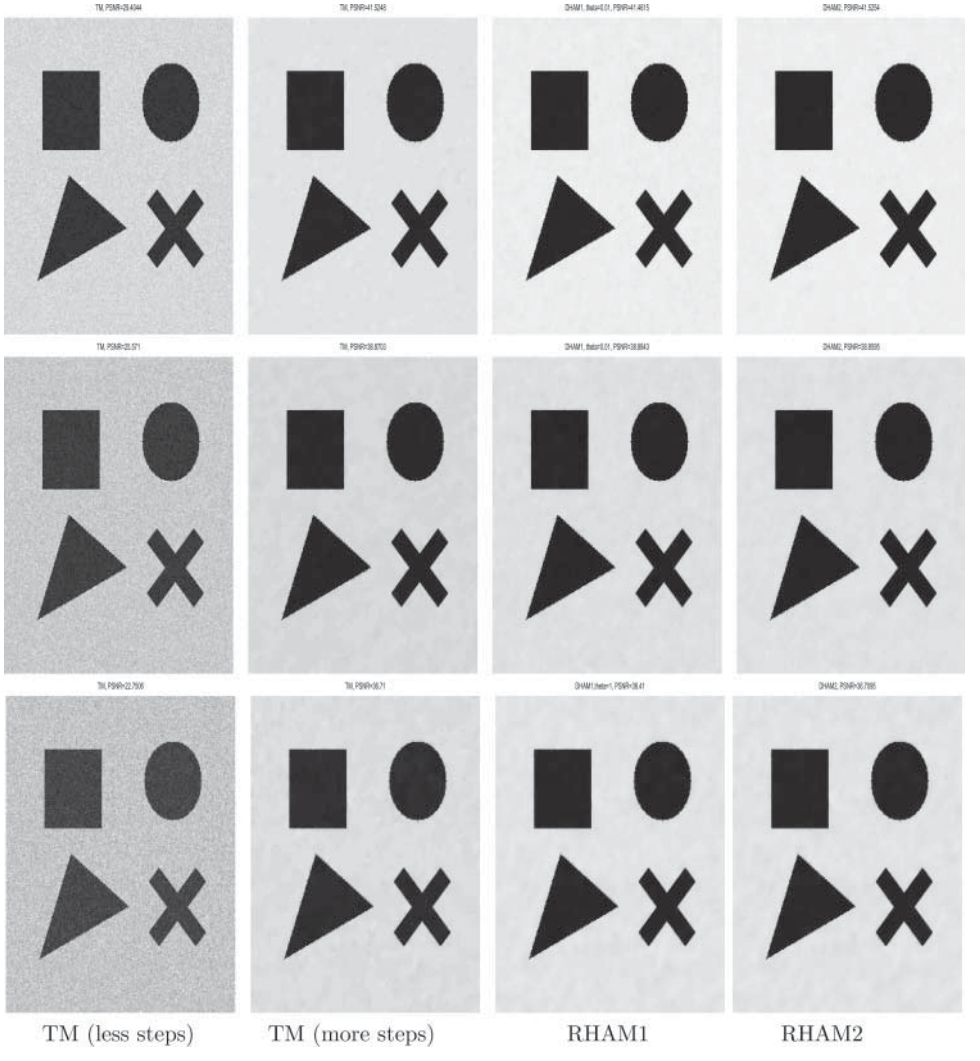

Figure 3. Test Set-1 Example 3. From the top row to the bottom, 10%, 15% and 20% random noise has been added to the image. For each row, the first image shows the result obtained with the TM method for the same number of iterations as the maximum iterations used for RHAM1 and RHAM2, the second image shows the result obtained with the TM method, and the third and fourth images show the results obtained for RHAM1 and RHAM2.

Figure 4. Test Set-1 Example 4. From the top row to the bottom, 10%, 15% and 20% random noise has been added to the image. For each row, the first image shows the result obtained with the TM method for the same number of iterations as the maximum iterations used for RHAM1 and RHAM2, the second image shows the result obtained with the TM method, and the third and fourth images show the results obtained for RHAM1 and RHAM2.

formulas:

$$\text{SNR} = 10\log_{10}\left(\frac{\sum_{(i,j)}(u_{i,j})^2}{\sum_{(i,j)}(u_{i,j}-v_{i,j})^2}\right) \quad \text{and} \quad \text{PSNR} = 10\log_{10}\frac{255^2}{\text{RMSE}}$$

$$\text{with} \quad \text{RMSE} = \sqrt{\frac{1}{n_1 n_2}\sum_{(i,j)}(u_{i,j}-v_{i,j})^2}, \tag{50}$$

where $u_{i,j}$ is the value of the true image at the grid point (i,j), $v_{i,j}$ is the value of the restored image at the grid point (i,j), $n_1 n_2$ is the total number of pixels (the smaller the SNR, the greater the noise). Since in real applications required to measure the difference between the restored images, using *PSNR* is more convenient. This is a relative measurement and can be used to compare two different denoising processes.

In the case of image segmentation, the relative residual has been used as an indicator of reaching the boundaries.

5.1 *Test set1: results for the denoising problem using the RHAM method*

In this subsection, we test our algorithms on denoising in several images with an intensity range [0,255] which includes real-life and artificial images. The image size shown is 256×256 and parameter values $\beta = 10^{-6}$, $\Delta t = 0.01$, $\hbar = -1$ and maxit $= 1000$ have been fixed. We repeated the same experiments for images of size 512×512 and obtained the same results as concluded for the smaller size. Different images and two types of noise, random normal and random uniform, have been applied. In the case of random normal noise, we will show the results for 10%, 15% and 20% noise, while for the uniform noise 10%, 30% and 50% noise has been used and in both cases the variable λ varies at $0.1, 0.05$ and 0.01, respectively. Figures 1–4 show the results obtained with the two new methods described in Algorithms 2 and 3 above in comparison with the TM

Table 1. Test Set-1 required iterations for different methods with Gaussian noise applied to the image.

Image	Noise	TM	TM PSNR	RHAM1	RHAM2	RHAMs PSNR
Lena	10	872	**33.60**	48	50	**33.60**
	15	1524	**31.40**	63	65	**31.48**
	20	1653	**30.01**	67	69	**30.13**
Peppers	10	1112	**33.95**	52	53	**33.95**
	15	1581	**31.61**	65	67	**31.72**
	20	1721	**30.19**	68	71	**30.20**
Objects	10	1415	**41.52**	82	88	**41.52**
	15	1941	**38.87**	98	99	**38.88**
	20	1895	**36.71**	85	87	**36.71**
Rocket	10	1308	**40.72**	71	75	**40.80**
	15	1781	**38.15**	83	85	**37.93**
	20	1729	**36.01**	78	81	**36.01**

Note: Required iterations for different methods (with $\Delta t = 0.01$) to achieve the PSNR values obtained from RHAM1 and RHAM2 by use of stopping criteria $\text{PSNR}(u_n) < \text{PSNR}(u_{n-1})$. The fourth column gives the PSNR reached with the TM method, while the last column gives the maximum PSNR reached with the RHAMs.

Table 2. Test Set-1 CPU time for different methods with Gaussian noise applied to the image.

Image	Noise	TM	TM PSNR	RHAM1	RHAM2	RHAMs PSNR
Lena	10	80.42	**33.60**	18.01	17.16	**33.60**
	15	120.80	**31.40**	21.73	22.43	**31.48**
	20	151.04	**30.01**	23.46	23.68	**30.13**
Peppers	10	797	**33.95**	19.68	18.42	**33.95**
	15	120.43	**31.61**	22.74	23.25	**31.72**
	20	150.82	**30.19**	20.84	24.32	**30.20**
Objects	10	115.83	**41.52**	28.16	30.15	**41.52**
	15	152.49	**38.87**	34.27	33.08	**38.88**
	20	154.65	**36.71**	24.89	30.13	**36.71**
Rocket	10	111.79	**40.72**	24.18	26.37	**40.80**
	15	130.68	**38.15**	27.86	29.35	**37.93**
	20	114.59	**36.01**	27.58	27.84	**36.01**

Note: CPU time for different methods (with $\Delta t = 0.01$) to achieve the PSNR values obtained from RHAM1 and RHAM2 by use of stopping criteria $\text{PSNR}(u_n) < \text{PSNR}(u_{n-1})$. The fourth column gives the PSNR reached with the TM method, while the last column gives the maximum PSNR reached with the RHAMs.

method for Gauss white noise. Tables 1 and 2 show the comparison in terms of CPU time and number of iterations required for those methods in the case of random noise and Tables 3 and 4 show the results in case of uniform random noise, while Table 5 shows the results obtained from applying the AOS method for the same figures in the case of random normal noise with different Δt. From Tables 1–4 we can note a great speed increase with RHAMs in comparison with TM, while Table 5 shows that the AOS method is slow for small Δt and loses some accuracy if we increase Δt to 1. The bold numbers in the PSNR columns in Table 5 show that the PSNR of both AOS and RHAMs has been the same, the non-bold PSNRs show the maximum that the AOS method could achieve. In the experiments, it has been noted that when the level of noise increases, increasing the parameter θ for RHAM1 gives better results. For this reason in cases of high noise, we set $\theta = 1$ instead of $\theta = 0.01$ considered for low noise in RHAM1.

Other fast methods such as multigrid [5,40,41] can be considered for comparison. We applied multigrid for all the images considered in Figures 1–4 and for brevity we show only one of them (Figure 5). As shown in this figure the multigrid method gives equally good accuracy in comparison with AOS (PSNR = 33.32) as well as a fast converge of the method in a few v-cycles (CPU time = 20.61). On the other hand, we have to mention that the parameter β must

Table 3. Test Set-1 required iterations for different methods with uniform random distributed noise.

Image	Noise	TM	RHAM1	RHAM2	PSNR
Lena	10	1000	41	44	**36.08**
	30	717	719	43	**24.1585**
	50	2441	2450	86	**19.8548**
Peppers	10	127	127	19	**33.4051**
	30	784	778	45	**24.2216**
	50	2705	2713	93	**19.8986**
Objects	10	216	217	26	**34.0754**
	30	1016	1017	65	**24.5201**
	50	2881	2881	127	**20.142**
Rocket	10	207	207	25	**34.0573**
	30	950	951	59	**24.5216**
	50	2729	2730	116	**20.0786**

Note: Required iterations for different methods (with $\Delta t = 0.01$) to achieve the PSNR values obtained from RHAM1 and RHAM2 by use of stopping criterion $\text{PSNR}(u_n) < \text{PSNR}(u_{n-1})$.

Table 4. Test Set-1 CPU time for different methods with uniform random distributed noise.

Image	Noise	TM	RHAM1	RHAM2	PSNR
Lena	10	0.8268	0.9672	0.1716	**33.3785**
	30	5.382	5.1948	0.4212	**24.1585**
	50	19.4065	20.1553	0.8892	**19.8548**
Peppers	10	0.9984	1.092	0.1716	**33.4051**
	30	5.850	5.616	0.4212	**24.2216**
	50	20.2489	20.8105	0.7332	**19.8986**
Objects	10	1.6380	1.6380	0.234	**34.0754**
	30	7.8156	8.4864	0.5460	**24.5201**
	50	21.7309	23.1817	0.9828	**20.142**
Rocket	10	1.2948	1.6692	0.2496	**34.0573**
	30	6.9888	7.8624	0.4836	**24.5216**
	50	20.8885	22.1677	0.8736	**20.0786**

Note: CPU time for different methods (with $\Delta t = 0.01$) to achieve the PSNR values obtained from RHAM1 and RHAM2 by use of stopping criterion $\text{PSNR}(u_n) < \text{PSNR}(u_{n-1})$.

Table 5. Test Set-1: AOS with Gaussian noise applied to the image to be compared with Tables 1 and 2.

		$\Delta t = 0.01$			$\Delta t = 0.1$			$\Delta t = 1$		
Image	Noise	CPU	No	PSNR	CPU	No	PSNR	CPU	No	PSNR
Lena	10	86.04	867	33.44	9.18	91	33.36	1.32	13	33.26
	15	130.26	1321	31.31	12.41	127	31.30	1.66	17	31.17
	20	147.48	1484	29.93	15.21	152	29.90	1.90	18	29.84
Peppers	10	77.47	782	**33.95**	9.06	93	**33.95**	1.54	15	33.83
	15	109.04	1108	**31.72**	12.32	125	**31.72**	1.87	18	31.72
	20	126.31	1275	**30.20**	13.11	133	**30.20**	1.81	16	30.20
Objects	10	124.84	1264	**41.46**	15.24	153	**41.46**	41.04	109	41.04
	15	205.85	2083	**38.85**	24.64	249	**38.85**	4.55	45	3824
	20	226.31	2286	36.23	19.68	199	36.22	2.9	128	35.80
Rocket	10	113.17	1141	**40.74**	12.52	126	**40.74**	4.45	44	40.59
	15	163.55	1646	**37.87**	17.05	172	**37.87**	3.1	31	37.68
	20	163.87	1644	**35.49**	16.16	164	**35.49**	2.60	25	35.42

Notes: Required iterations for the AOS method (with $\Delta t = 0.01$, 0.1 and 1) to achieve the PSNR values obtained from RHAM1 and RHAM2. The bold numbers in the PSNR columns shows that the PSNRs of both AOS and RHAMs have been the same, the rest of non-bold PSNRs is the maximum that the AOS method could achieve.

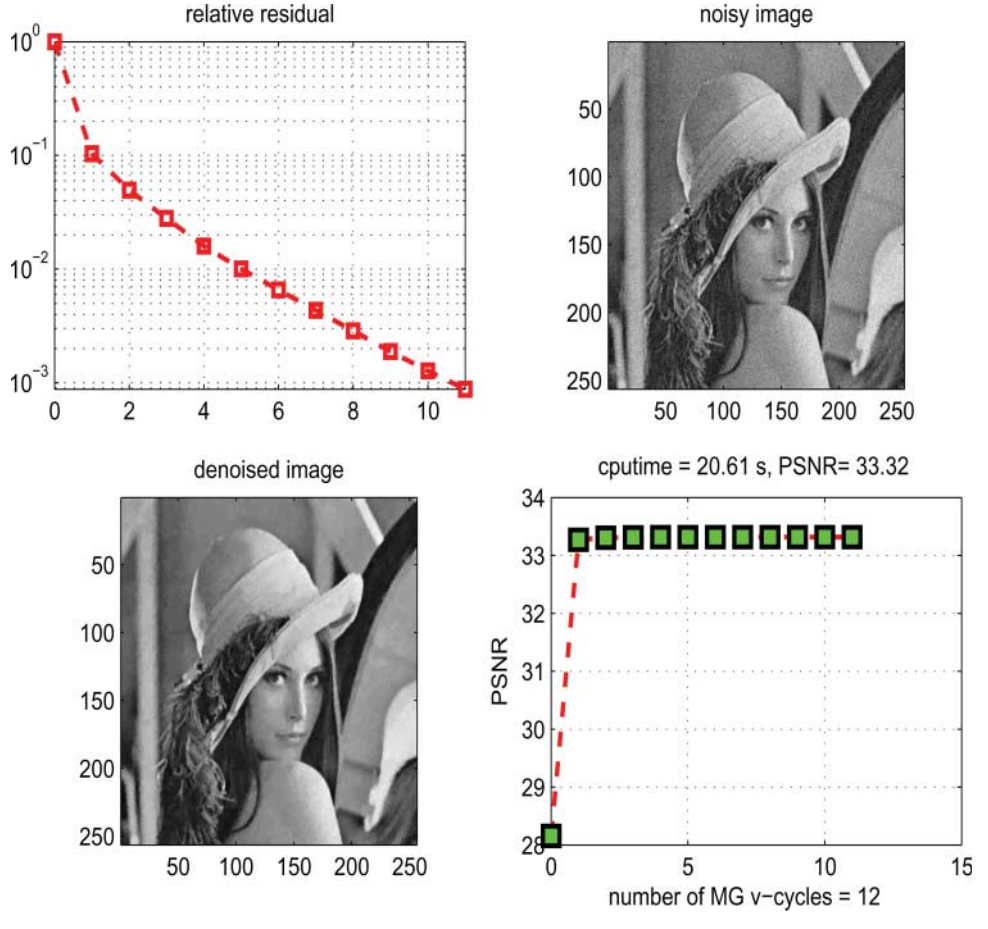

Figure 5. Test Set-1 Example 5. Denoising of the Lenna image with 10% random noise with multigrid. The first image shows the relative residual, the second image shows the noisy image, the third image is the obtained denoised image with multigrid method and the fourth image show the PSNR value for each v-cycle (colour online only).

Table 6. Test Set-2 comparison of RHAM with TM.

	CPU time			No. of iterations used		
$\Delta t = 0.01$	TM	RHAM1	RHAM2	TM	RHAM1	RHAM2
Figure 6	143.07	4.74	4.75	1287	29	32
Figure 7	22.45	4.24	4.82	205	25	33
Figure 8	20.03	2.19	2.21	186	9	9
Figure 9	97.83	6.70	7.50	929	46	53
Figure 10	49.48	7.22	6.45	467	49	47

Note: Comparison for CPU time recorded and the maximum iterations needed to have a given residual as shown in the respective picture between TM, RHAM1 and RHAM2 methods.

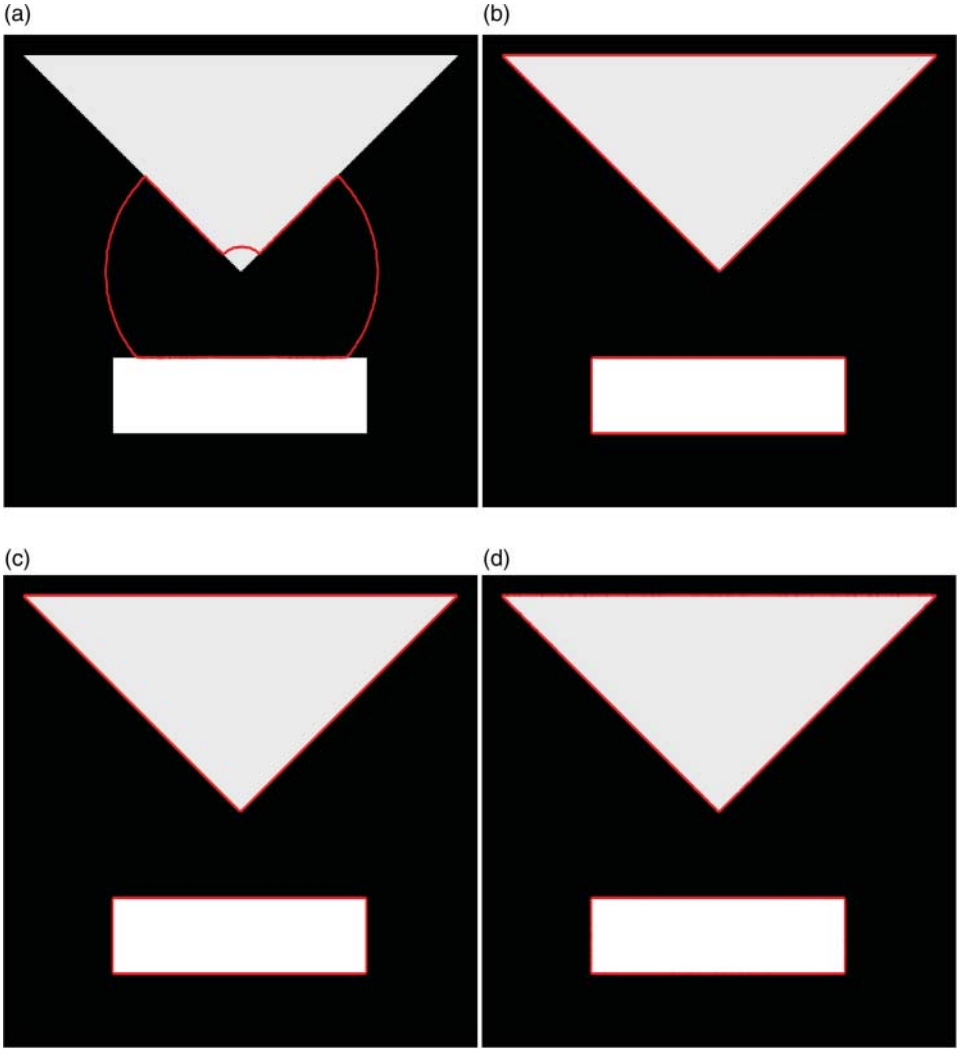

Figure 6. Test Set-2 Example 1. Successfully reached relative residual equal to 10^{-2} after 32 iterations for RHAMs and 1287 iterations for TM. (a) TM method result after 32 iterations, (b) HAM1 method result after 32 iterations, (c) HAM2 method result after 32 iterations and (d) TM method result after 1287 iterations (colour online only).

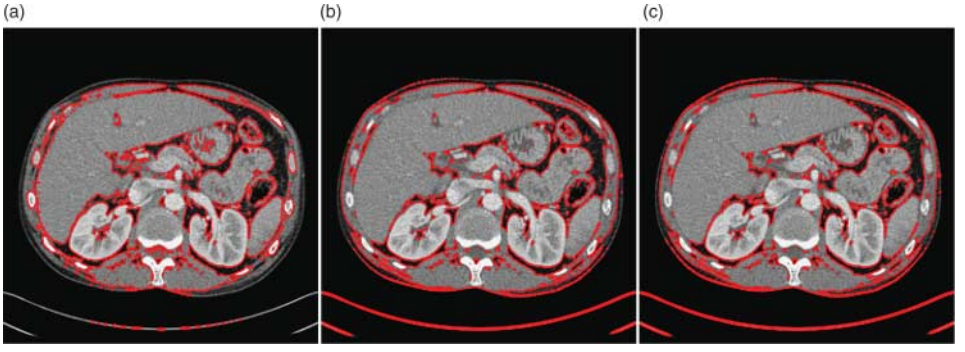

Figure 7. Test Set-1 Example 2. Successfully reached relative residual equal to 10^{-3} after 33 iterations for RHAMs and 205 iterations for TM. (a) TM method result after 33 iterations, (b) HAM1 method result after 33 iterations and (c) HAM2 method result after 33 iterations (colour online only).

Figure 8. Test Set-1 Example 3. Successfully reached relative residual equal to 10^{-2} after 9 iterations for RHAMs and 186 iterations for TM. (a) TM method result after 9 iterations, (b) HAM1 method result after 9 iterations and (c) HAM2 method result after 9 iterations (colour online only).

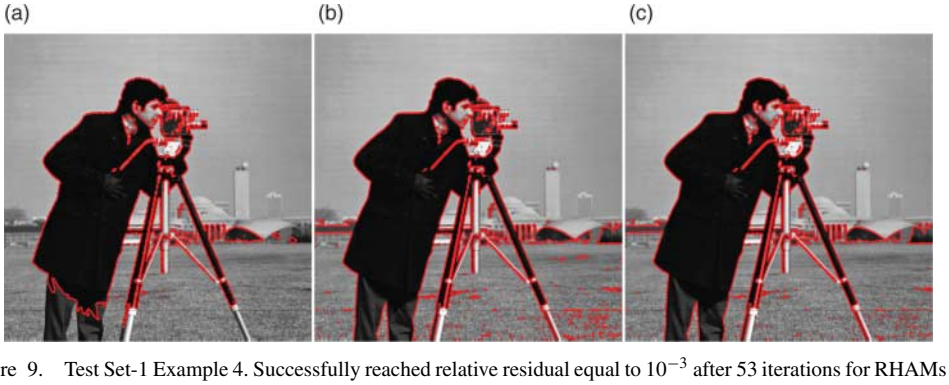

Figure 9. Test Set-1 Example 4. Successfully reached relative residual equal to 10^{-3} after 53 iterations for RHAMs and 929 iterations for TM. (a) TM method result after 53 iterations, (b) HAM1 method result after 53 iterations and (c) HAM2 method result after 53 iterations (colour online only).

be larger than 10^{-2} while applying the multigrid method due to convergence. Moreover, we can note from Table 2 that RHAM is compatible with the multigrid method by giving a better accuracy (PSNR = 33.60) without restriction for the parameter β (for RHAM $\beta = 10^{-6}$).

5.2 *Test set2: experimental results for segmentation using the RHAM method*

In the following experiments, the parameters θ, Δt, μ, λ_1, λ_2, h (step size) and $\hbar$ have been fixed as follows: $\theta = 1$, $\Delta t = 0.01$, $\mu = 1$, $\lambda_1 = \lambda_2 = 50$, $h = 1$ and $\hbar = -1$. The size of the

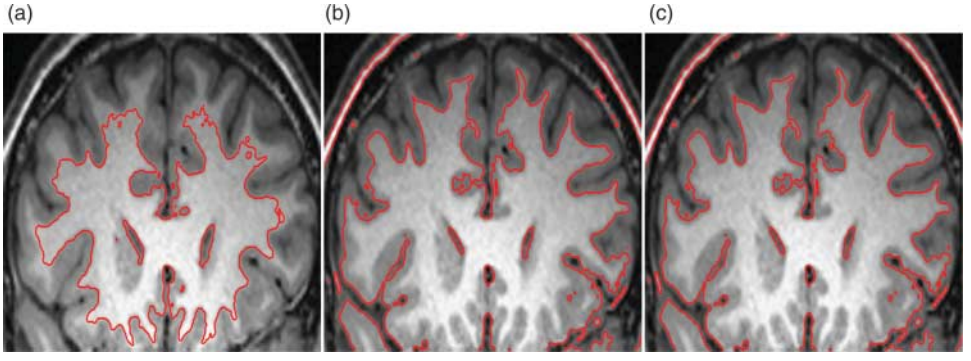

Figure 10. Test Set-1 Example 5. Successfully reached relative residual equal to 10^{-3} after 47 iterations for RHAMs and 467 iterations for TM. (a) TM method result after 47 iterations, (b) HAM1 method result after 47 iterations and (c) HAM2 method result after 47 iterations (colour online only).

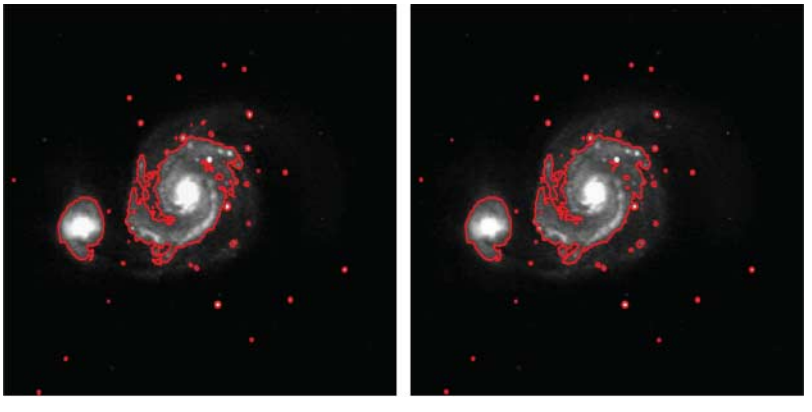

Figure 11. Test Set-1 Example 6. Successfully segmented after 59 iteration for RHAM2 and 1558 for the AOS method to reach relative residual equal to 10^{-3} (colour online only).

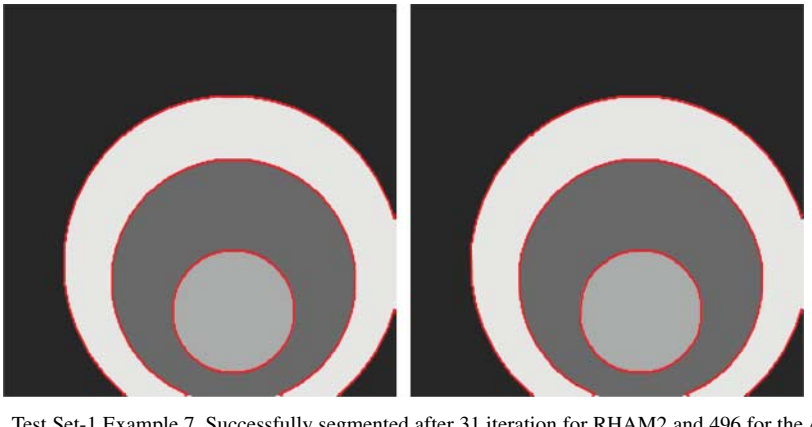

Figure 12. Test Set-1 Example 7. Successfully segmented after 31 iteration for RHAM2 and 496 for the AOS method to reach relative residual equal to 10^{-3} (colour online only).

Table 7. Test Set-2 comparison of RHAM with AOS.

	$\Delta t = 0.01$		$\Delta t = 0.1$		$\Delta t = 1$	
CPU time	AOS	RHAM1	AOS	RHAM1	AOS	RHAM1
Figure 11	593.73	22.67	47.53	10.17	13.87	2.98
Figure 12	175.00	12.05	53.03	3.03	6.32	2.50

Note: CPU time recorded for the iteration needed to have a given residual as shown in the respective picture.

Table 8. Test Set-2 comparison of RHAM with AOS.

	$\Delta t = 0.01$		$\Delta t = 0.1$		$\Delta t = 1$	
No. of iterations used	AOS	RHAM1	AOS	RHAM1	AOS	RHAM1
Figure 11	1558	59	139	39	40	4
Figure 12	496	31	152	4	17	2

Note: Maximum iterations needed to have a given residual as shown in the respective picture.

image considered in the following is $n = 256$ and the initial level set is placed in the centre of the image and the radius is 50. To stop the program, the relative residual 10^{-2} or 10^{-3} has been used. Table 6 shows the comparison between the TM method and RHAM1 and RHAM2 in CPU time and maximum iterations needed to have the given residual as shown in each of the pictures. It can be easily noted that both RHAM1 and RHAM2 are at least 10 times faster than the TM method.

It is well known that for the AOS method Δt can be chosen sufficiently large that we get fast convergence. In the following experiments, we show the results obtained comparing the RHAM1 with the AOS method. Figures 11 and 12 show the results obtained for $\Delta t = 0.01$, while Tables 7 and 8 show the comparison between the methods for different values of Δt. It can be noted that the RHAM method is at least two times faster than the AOS method.

6. Conclusions

In this paper, we proposed a discrete RHAM to obtain numerical solutions for image denoising and segmentation which are important in practical problems in image processing. The experimental results show that the method is effective in giving fast solutions for denoising and segmentation, and at the same time reliable and delivers higher PSNR values compared with other methods. Numerical tests demonstrate that RHAM shows great speed compared with TM and is faster than the AOS method. Our results in denoising and segmentation suggest that RHAM may be used to potentially solve other nonlinear PDEs arising from image processing models, which will be our future work.

Notes

1. The operator $\mathcal{L}_1(u)$ in Equation (11) is a general case of the linear operator applied by Liao [24]
2. In this paper, we consider the most common additive noise such as random Gaussian noise with mean 0 and standard deviation σ and uniform distributed noise
3. Without no need of solving a system.
4. $f_0'(\phi_0)$ has been calculated by the chain rule differentiation formula and c_1' and c_2' in Equations (44) and (45) has been differentiated with respect to q.

References

[1] S. Abbasbandy, *Solitary wave solutions to the Kuramoto–Sivashinsky equation by means of the homotopy analysis method*, Nonlinear Dyn. 52 (2008), pp. 35–40.
[2] S. Abbasbandy and F. Zakaria, *Soliton solutions for the fifth-order kdv equation with the homotopy analysis method*, Nonlinear Dyn. 51(1) (2007), pp. 83–87.
[3] G. Adomian, *A review of the decomposition method and some recent results for nonlinear equations*, Comput. Math. Appl. 21 (1991), pp. 101–127.
[4] I.V. Andrianov and V.V. Danishevky, *Asymptotic approach for non-linear periodical vibrations of continuous structures*, J. Sound Vibration 249(3) (2002), pp. 465–481.
[5] N. Badshah and K. Chen, *Multigrid method for the Chan–Vese model in variational segmentation*, CiCP 4(2) (2008), pp. 294–316.
[6] C. Brito-Loeza and K. Chen, *Multigrid algorithm for high order denoising*, SIAM J. Imaging Sci. 3(3) (2010), pp. 363–389.
[7] C. Brito-Loeza and K. Chen, *On high-order denoising models and fast algorithms for vector-valued images*, IEEE Trans. Image Proc. 19 (2010), pp. 1518–1527.
[8] V. Caselles, R. Kimmel, and G. Sapiro, *Geodesic active contours*, Proceedings of the 5th International Conference on Computer Vision, Boston, MA, 1995, pp. 694–699.
[9] V. Caselles, R. Kimmel, and G. Sapiro, *Geodesic active contours*, Int. J. Comput. Vis. 22(1) (1997), pp. 61–79.
[10] A. Chambolle and P.L. Lions, *Image recovery via total variation minimization and related problems*, Numer. Math. 76 (1997), pp. 167–188. doi:10.1007/s002110050258.
[11] R.H. Chan and K. Chen, *Multilevel algorithm for a Poisson noise removal model with total-variation regularization*, Int. J. Comput. Math. 84 (2007), pp. 1183–1198.
[12] T. Chan and L. Vese. *An active contour model without edges*, International Conference on Scale-Space Theories in Computer Vision, Greece, 1999, pp. 141–151.
[13] T.F. Chan and L.A. Vese, *Active contours without edges*, IEEE Trans. Image Process. 10(2) (2001), pp. 266–277.
[14] T.F. Chan, G.H. Golub, and P. Mulet, *A nonlinear primal-dual method for total variation-based image restoration*, SIAM J. Sci. Comput. 20 (1999), pp. 1964–1977.
[15] T.F. Chan, B.Y. Sandberg, and L.A. Vese, *Active contours without edges for vector-valued images*, J. Visual Commun. Image Represent. 11 (2000), pp. 130–141.
[16] N. Chumchob, K. Chen, and C. Brito-Loeza, *A new variational model for removal of combined additive and multiplicative noise and a fast algorithm for its numerical approximation*, Int. J. Comput. Math. 90(1) (2013), pp. 140–161.
[17] M. Dehghan, J. Manafian, and A. Saadatmandi, *Solving nonlinear fractional partial differential equations using the homotopy analysis method*, Numer. Methods Partial Differential Equations 26(2) (2010), pp. 448–479.
[18] T. Hayat, N. Ahmed, M. Sajid, and S. Asghar, *On the MHD flow of a second grade fluid in a porous channel*, Comput. Math. Appl. 54(3) (2007), pp. 407–414.
[19] M. Hintermuller and W. Ring, *An inexact Newton-CG-type active contour approach for the minimization of the Mumford–Shah functional*, J. Math. Imaging Vis. 20 (2004), pp. 19–42.
[20] H. Jafari and S. Seifi. *Solving a system of nonlinear fractional partial differential equations using homotopy analysis method*, Commun. Nonlinear Sci. Numer. Simul. 14(5) (2009), pp. 1962–1969.
[21] M. Kass, A. Witkin, and D. Terzopoulos, *Snakes: Active contour models*, Int. J. Comput. Vis. 1(4) (1988), pp. 321–331.
[22] S.J. Liao, *The proposed homotopy analysis technique for the solution of nonlinear problems*, Ph.D. Thesis, Shanghai Jiao Tong University, 1992.
[23] S.J. Liao, *A challenging nonlinear problem for numerical techniques*, J. Comput. Appl. Math. 181 (2005), pp. 467–472.
[24] S.J. Liao, *Beyond Perturbation: Introduction to the Homotopy Analysis Method*, Chapman & Hall/CRC, Boca Raton, FL, 2003.
[25] S.J. Liao. *On the homotopy analysis method for nonlinear problems*, Appl. Math. Comput. 147(2) (2004), pp. 499–513.
[26] S.J. Liao, *A new branch of solutions of boundary-layer flows over a permeable stretching plate*, Int. J. Non-Linear Mech. 42(6) (2007), pp. 819–830.
[27] S.J. Liao, *Notes on the homotopy analysis method: Some definitions and theorems*, Commun. Nonlinear Sci. Numer. Simul. 14(4) (2009), pp. 983–997.
[28] S.J. Liao, *On the relationship between the homotopy analysis method and Euler transform*, Commun. Nonlinear Sci. Numer. Simul. 15(6) (2010), pp. 1421–1431.
[29] S.J. Liao and Y. Tan, *A general approach to obtain series solutions of nonlinear differential equations*, Stud. Appl. Math. 119 (2007), pp. 297–355.
[30] M. Lysaker and X.C. Tai, *Iterative image restoration combining total variation minimization and a second-order functional*, Int. J. Comput. Vis. 66 (2006), pp. 5–18. doi:10.1007/s11263-005-3219-7.
[31] M. Lysaker, S. Osher, and X.C. Tai, *Noise removal using smoothed normals and surface fitting*, IEEE Trans. Image Process. 13(10) (2004), pp. 1345–1357.
[32] 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), pp. 387–405.
[33] D. Mumford and J. Shah, *Optimal approximation by piecewise smooth functions and associated variational problems*, Commun. Pure Appl. Math. 42 (1989), pp. 577–685.

[34] J.A. Murdock, *Perturbations: Theory and Methods*, John Wiley and Sons, New York, 1991.
[35] A.H. Nayfeh, *Perturbations Methods*, John Wiley and Sons, New York, 2000.
[36] L. Rada and K. Chen, *A new variational model with dual level set functions for selective segmentation*, Commun. Comput. Phys. 12(1) (2012), pp. 261–283.
[37] L.I. Rudin and S. Osher, *Total variation based image restoration with free local constraints*, Proceedings ICIP, IEEE International Conference on Image Processing, Austin, TX, Vol. 1, 1994, pp. 31–35.
[38] L.I. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Phys. D 60 (1992), pp. 259–268.
[39] M. Sajid, T. Hayat, and S. Asghar, *Non-similar solution for the axisymmetric flow of a third-grade fluid over a radially stretching sheet*, Acta Mech. 189 (2007), pp. 193–205.
[40] Jo. Savage and K. Chen, *An improved and accelerated non-linear multigrid method for total-variation denoising*, Int. J. Comput. Math. 82(8) (2005), pp. 1001–1015.
[41] J. Savage and K. Chen, *On multigrids for solving a class of improved total variation based staircasing reduction models*, in *Image Processing Based on Partial Differential Equations Mathematics and Visualization*, X-C. Tai, K.-A. Lie, T. Chan, and S. Osher, eds., Springer, Berlin, Heidelberg, 2007, pp. 69–94.
[42] J.A. Sethian. *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision and Material Science*, Cambridge University Press, Cambridge, 1996.
[43] A. Tsai, A.J. Yezzi Jr., and A.S. Willsky, *Curve evolution implementation of the Mumford–Shah functional for image segmentation, denoising, interpolation*, IEEE Trans. Image Process. 10(8) (2001), pp. 1169–1186.
[44] Y.-L. You and M. Kaveh, *Fourth-order partial differential equations for noise removal*, IEEE Trans. Image Process. 9(10) (2000), pp. 1723–1730.
[45] L.A. Vese, *Variational Methods in Image Processing*, Chapman & Hall, CRC Press, Boca Raton, FL, 2012.
[46] L.A. Vese and T.F. Chan, *A multiphase level set framework for image segmentation using the Mumford and Shah model*, Int. J. Comput. Vis. 50(3) (2002), pp. 271–293.
[47] C.R. Vogel, *A multigrid method for total variation-based image denoising*, in *Computation and Control IV*, Birkhauser, Boston, MA, 1995, pp. 323–331.
[48] C.R. Vogel and M.E. Oman, *Iterative methods for total variation denoising*, SIAM J. Sci. Comput. 17 (1996), pp. 227–238.
[49] J. Weickert, B.M. Ter Haar Romeny, and M.A. Viergever, *Efficient and reliable schemes for nonlinear diffusion filtering*, IEEE Trans. Image Process. 7(3) (1998), pp. 398–410.
[50] J. Weickert, B.M. Romeny, and M.A. Viergever, *Efficient and reliable schemes for nonlinear diffusion filtering*, IEEE Trans. Image Process. 7(3) (1998), pp. 398–410.
[51] F. Yang, K. Chen, and B. Yu, *Homotopy method for a mean curvature-based denoising model*, Appl. Numer. Math. 62(3) (2012), pp. 185–200.
[52] S.-P. Zhu, *An exact and explicit solution for the valuation of American put options*, Quant. Finance 6(3) (2006), pp. 229–242.
[53] W. Zhu and T.F. Chan, *Image denoising using mean curvature*, Unpublished article, 2008. Available at http://www.math.nyu.edu/∼wzhu/.
[54] W. Zhu and T.F. Chan, *Image denoising using mean curvature of image surface*, SIAM J. Imaging Sci. 5 (2012), pp. 1–32.
[55] H. Zhu, H. Shu, and M. Ding, *Numerical solutions of partial differential equations by discrete homotopy analysis method*, Appl. Math. Comput. 216(12) (2010), pp. 3592–3605.

Appendix

To answer a simple question such as how does the RHAM method perform in comparison with HAM we consider two simple examples.

Considering the following nonlinear second-order boundary value problem:

$$y''^3 - 6y(x) - 2x^3 = 0,$$

$$y(1) = 2, y(2) = \tfrac{5}{2};$$

it can be found that the exact solution has the form $y(x) = x + 1/x$. Starting with the initial condition $y_0(x) = x^2 - \frac{5}{2}x + \frac{7}{2}$, it can be shown, Figure A1, that the RHAM obtained in four restarted iterations reaches a good approximation to the exact solution.

As a second example, we consider the first-order boundary value problem:

$$y' - y^2(x) - 1 = 0, \quad y(0) = 0,$$

with exact solution in the form $y(x) = \tanh(x)$.

Figure A2 shows the result of RHAM obtained in five restarted iterations in comparison with high-order HAM up to 10th order, for the considered initial guess $y_0(x) = 1 - 1/(1 + x)$.

Other examples have been tested and concluded that the RHAM needs only a few reinitialization iterations to reach the same performance with high-order HAM.

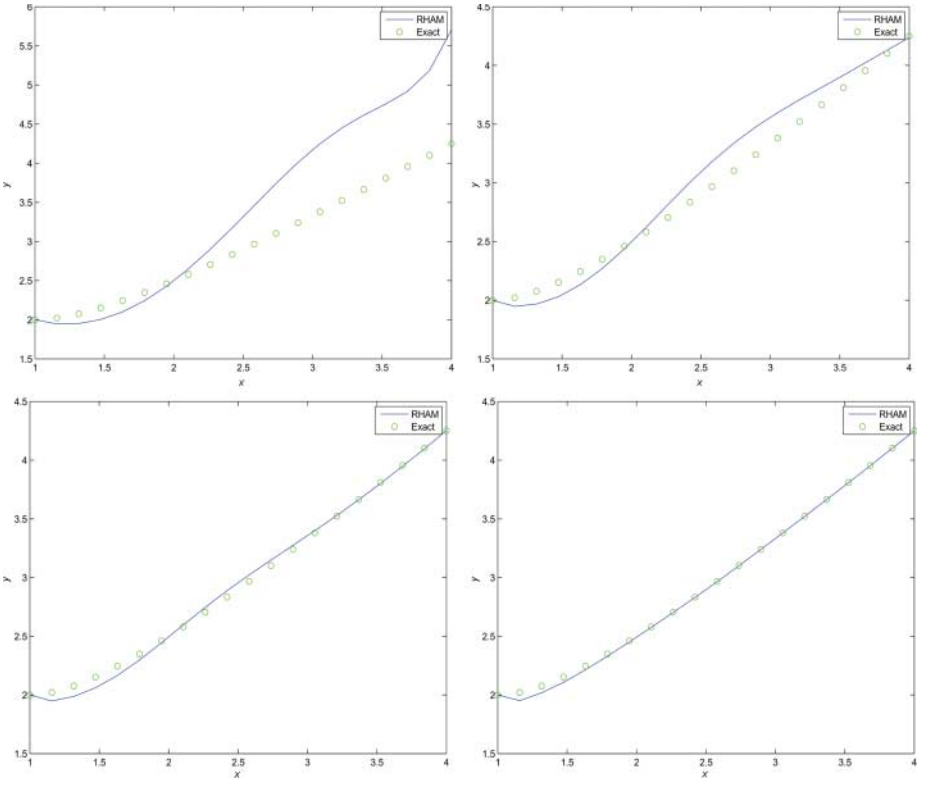

Figure A1. From the top left to the right, evolution of RHAM reaches great accuracy in comparison with the exact solution. The blue line shows the result of RHAM and the green line the exact solution. These results are the same with HAM including more expended terms (HAM needs to be at least eighth-order expended) (colour online only).

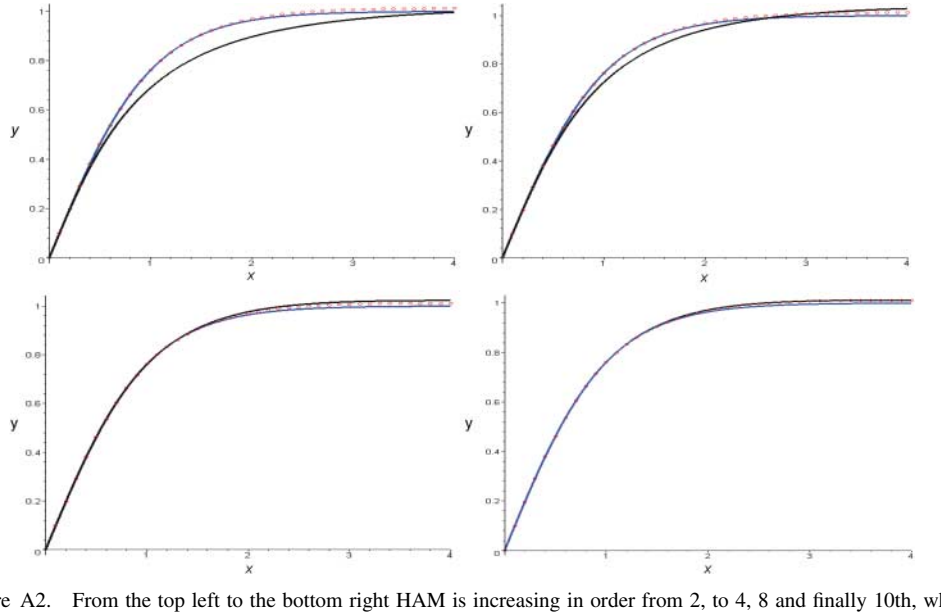

Figure A2. From the top left to the bottom right HAM is increasing in order from 2, to 4, 8 and finally 10th, which reaches the same accuracy as second order RHAM with 5 re-initializations for 10th order HAM. The blue line shows the exact solution, the red line shows RHAM for $m = 2$ with 5 re-initializations and the black line shows HAM. (colour online only).