Semi-adaptive, convex optimisation methodology for image denoising

T. Kärkkäinen and K. Majava

Abstract: An optimisation methodology based on a semi-adaptive, convex (SAC) formulation for the image denoising problem is proposed for recovering both sharp edges and smooth subsurfaces from a given noisy image. Basic steps to realise an image denoising algorithm with proper restoration properties and practical computational efficiency with automatic determination of free parameters are described. A set of example images is used to illustrate the proposed approach.

1 Introduction

Image denoising is a fundamental task in image processing. In various applications of computer vision, image processing is usually started by removing or reducing noise and other distortions from the digital image. In many applications, especially if edge detection or segmentation is required, the crucial task in the denoising process is the preservation of sharp edges [1, 2]. A common drawback of standard denoising methods, like filtering using Fourier or wavelet transforms or statistical methods [3, 4], is that they are linear and hence smear sharp edges. Recently, several adaptive non-linear denoising methods based on wavelets have been proposed to better preserve the edges of the image ([5, 6] and the references therein). Other efforts in this direction have been made, for example in [7], where an adaptive median filter was proposed and in [8], where the method presented was based on the psychophysical phenomenon of human visual contrast sensitivity.

In this paper, we discuss optimisation-based techniques for image denoising. They have proven to be very efficient in preserving edges, especially the method based on total (or bounded) variation (TV) [9]. A well-known drawback of this method is, however, that the denoised image contains a staircase-like structure, which is not optimal for images with smooth subsurfaces. A lot of effort has recently been made in order to decrease the staircase effect of the TV method [10-15], and that is also the purpose of this paper.

An optimisation methodology based on a semi-adaptive, convex (SAC) formulation for denoising model was localised and a digital TV filter was solving the TV problem [16-18]. In [19], the global TV denoising model was localised and a digital TV filter was proposed. The TV term does not penalise discontinuities in u, thus allowing one to recover sharp edges of the original image. However, the theoretical analysis in [20, 21] and numerical experiments, [11, 12, 16, 22] show that the denoised image due to the TV formulation contains a staircase-like structure, which is not optimal for images with smooth subsurfaces. On the other hand, having $s > 1$ recovers smooth surfaces better but smears sharp edges of the image [11, 23, 24]. Hence, some form of adaptivity is needed for an improved image restoration capability.

Adaptive optimisation formulations for image restoration have been considered [10-14]. As soon as adaptivity is required, formulations tend to become much more complicated. Adaptive formulations are often non-convex or non-smooth (or even both), which makes the solution process complicated, and in many cases the number of unknowns and free parameters in these formulations is increased. Closest to our approach is the adaptive formulation [10], where the idea was to use a TV-like regularisation ($s = 1$ in (1)) near edges, smooth regularisation ($s = 2$) in flat regions, and values $1 < s < 2$, between. The exponent $s$ was chosen to be a gradient-driven function $s = s(\|\nabla u\|)$, which made the formulation non-convex and hence its solution non-unique.

The SAC method proposed in this paper is a simplification of the method [10]. The SAC formulation is not purely adaptive but always strictly convex and sufficiently smooth that well-known solution methods, especially the conjugate gradient method, can be applied as part of the solution process. The principal part of the algorithm, however, relies on the fast active-set method for solving the TV-regularised image restoration problem, as described in [16]. Concerning the proposed approach, we also emphasise that using the heuristic determination of the regularisation parameter, as
suggested and tested in [24, 25], we do not need any a priori information concerning the amount of (Gaussian) noise contained in the given original image. Other suggestions for choosing the regularisation parameter have been discussed [26].

The SAC method describes basic steps to realise an image denoising algorithm with proper restoration properties and computational efficiency that allows restoration of practical sizes of images. The proposed substeps can be modified for different application areas when doing practical restoration, so that instead of a single method, we obtain a methodology for image denoising. In this paper, the SAC algorithm is developed for two-dimensional images. Some of the underlying ideas were presented and successfully tested for 1-D images in [27].

2 Preliminaries

In this Section, the basic formulation for the image denoising problem with fixed $s$ is discussed, because it forms the basis for the SAC formulation. For this purpose, let us consider the two-dimensional image denoising problem [24] in a form

$$\min_{w \in H_0^2(\Omega)} \frac{1}{2} \int_{\Omega} |u - z|^2 \, dx + \frac{\mu}{2} \int_{\Omega} |\nabla w|^2 \, dx + \frac{\alpha}{2} \int_{\Omega} |\nabla z|^2 \, dx$$

where $1 \leq s \leq 2$. Here $\Omega \subset \mathbb{R}^2$ is a rectangle, $z$ is the noisy data, and $\alpha > 0$ is a regularisation parameter. Further, the term $(\mu)/2 \int_{\Omega} |\nabla u|^2 \, dx$ for $0 < \mu \ll g$ ensures coercivity of the cost functional in $H_0^2(\Omega)$ and yields the unique solvability of problem (2) in this space [16, 22]. In the discrete case, also $\mu = 0$ allows a unique solution in the Euclidean norm.

The $s$-regularisation term $\psi_s(\mathbf{t}) = (1/2) \int_{\Omega} |\mathbf{t}|^2 \, dx$ is classically differentiable for $s > 1$. For $s = 2$, $\psi_2(\mathbf{t})$ is Lipschitz continuous but for $1 < s < 2$, $\psi_s(\mathbf{t})$ is only Hölder continuous, so that convergence assumptions of gradient-descent methods, especially the conjugate gradient (CG) method, are not satisfied for $1 < s < 2$. However, problems with the smoothness of the function appear only around point 0. Moreover, the $s$-regularisation $\psi_s(\mathbf{t})$ is strictly convex for $s > 1$ and convex for $s = 1$.

Consider the discretisation of problem (2) via the finite-difference method. We assume that the discrete domain is a rectangle $(0, l_1) \times (0, l_2)$ with equidistant division into subintervals with $n_1$ pixels into the x-direction and $n_2$ into the y-direction, respectively. In practice, when an $n_1 \times n_2$ image is given, we let $h = 1/(\min(n_1, n_2) + 1)$ and take $l_1 = h(n_1 + 1), l_2 = h(n_2 + 1)$. Because only the discrete case is considered in the rest of the paper, we let $u, z, \lambda_1, \lambda_2$ denote vectors in $R^{n_1 \times n_2}$, whose components correspond to the mesh points. Then $H_0^2(\Omega)$ in (2) is replaced with $R^{n_1 \times n_2}$

$\min_{w \in R^{n_1 \times n_2}} J_s(u) = \frac{1}{2} \left( (u - z)^T (u - z) + \frac{\mu}{2} \nabla u^T K \nabla u \right) + \frac{\mu}{2} \nabla u^T K \nabla u$ 

where $K = \frac{1}{s} \sum_{i=1}^{n_2} \frac{1}{s_i} |\nabla u_i|^{s_i}$

The discrete approximation of problem (2) is then defined as

$$\min_{w \in R^{n_1 \times n_2}} J_s(u) = \frac{1}{2} \left( (u - z)^T (u - z) + \frac{\mu}{2} \Delta u^T K \Delta u \right) + \frac{\mu}{2} \Delta u^T K \Delta u$$

and $1 \leq s \leq 2$. For all $\mu \geq 0$, $J_s(u)$ is strictly convex and, for $s > 1$, it is also differentiable in $R^{n_1 \times n_2}$. In the latter case, the two identities

$$u^* - z + \mu K u^* + g \nabla \bar{z}^* = 0 \quad (\lambda^* , \bar{z}^*)$$

for $l = 1, 2$, constitute the necessary and sufficient optimality condition for the solution $u^*$ of problem (3). For $s = 1$, $J_s(u)$ is non-differentiable, and a subgradient $\xi \in \partial J_s(u)$ is defined by

$$\xi = u^* - z + \mu K u^* + g \nabla \bar{z}^*$$

where $|\lambda^*| < 1$ for $\partial J_s(u) = 0$. Now, $0 \in \partial J_s(u)$ is the necessary and sufficient optimality condition for problem (3) with $s = 1$.

3 SAC semi-adaptive, convex approach

The basic idea of the SAC method is similar to [10]: we use smooth regularisation in the presumed smooth parts of an image and a TV-type regularisation for edges. Unlike in [10], however, the division of the given image into differently regularised parts is made explicitly.

In the SAC method, we solve the following strictly convex optimisation problem

$$\min_{w \in R^{n_1 \times n_2}} J_s(u) = \frac{1}{2} \left( (u - z)^T (u - z) + \frac{\mu}{2} \Delta u^T K \Delta u \right) + \frac{\mu}{2} \Delta u^T K \Delta u$$

where $1 \leq s \leq 2$. The only difference between problems (6) and (3) is that instead of fixed $s$, we have a set of fixed values $\{s_i\}, i = 1, \ldots, n_1 n_2 - 1$, which are determined by using a reference solution $\bar{u}$. The optimality condition for problem (6) is of the form (4) – (5), where $s$ is replaced by $s_i = \lambda_i$ for $(\lambda_i^*, \bar{z}^*)$. The actual SAC method consists of the following three steps:

Step 1. Compute a reference solution $\bar{u}$ and fix the regularisation parameter $g$ in (6).

Step 2. Determine $\{s_i\}, i = 1, \ldots, n_1 n_2 - 1$, using $\bar{u}$.


In the following Sections, we explain how the three steps of the SAC method have been realised.

3.1 Reference solution $\bar{u}$ and determination of $g$

3.1.1 Reference solution: As was mentioned in the Introduction, the SAC method performs a kind of post-processing on the TV result. Hence, as a reference solution $\bar{u}$, we use the result $u_{TV}$ obtained from the TV formulation (3) for $s = 1$. Our choice of $\bar{u}$ is based on the observation that $u_{TV}$ contains a lot of valuable information of the true
image. More precisely, we associate the following interpretation to the result $u^*_{TV}$ possessing a staircase-like structure: the size of the gradient is related to the probability of having an edge in the true image, i.e. in smooth areas of the true image, $u^*_{TV}$ contains only small stairs, whereas proper edges of the true image coincide with large jumps. Notice that $u^*_{TV}$ was used as a reference solution also in [28], where the semi-adaptivity was focused on making the regularisation parameter $g$ vary within the TV framework as $\sum_{i=1}^{n} g_i |(D_i u)|$.

We solve the TV problem (3) for $s = 1$ using the active-set method (AS) proposed in [16]. For the initialisation of AS, we use $u_0 = z$.

3.1.2 Determination of $g$: In [25], a heuristic approach for determining an appropriate value of the regularisation parameter $g$ in the TV case was presented. The approach was successfully applied also to other simple image denoising formulations in [24]. The underlying idea is that we should choose $g$ to be the one giving the smallest value of the root-mean-squared (RMS) reconstruction error

\[ e(u') = \frac{1}{1/0,2} \sum_{i=1}^{n} (u' - z^*)^2 \]

where $z^*$ denotes the true image and $u'$ the solution of the TV problem (3) with $s = 1$. Of course, the RMS reconstruction error cannot be computed in real applications, where the true image is not known. The observation that the regularisation part $r(g) = g |D_i u|$ in (3) correlates with the RMS reconstruction error allows us to pick $g$ that yields the smallest value of $r(g)$. The following algorithm realises a fast search of $g$ according to this suggestion on an a priori selected set of $17$ possible values of $g$, which are stored in an increasing order in vector $G$. The algorithm is illustrated in Fig. 1.

Algorithm 3.1: Selection of $g$ in the TV-problem

(i) Set $G = \{G(3), G(7), G(11), G(15)\}$ and solve the TV problem for these values. Choose $i^*$ as $\text{argmin}_i r(g_i)$, $i = 1, \ldots, 4$.

(ii) Solve the TV problem for $g_5 = G(i^* - 2)$ and $g_6 = G(i^* + 2)$.

(iii) If $r(g_5) < r(g_6)$, set $g_7 = G(i^* - 1)$; otherwise set $g_7 = G(i^* + 1)$. Solve the TV problem for $g_7$.

(iv) Choose $g$ as the parameter that minimises $r(g_i)$, $i = 1, \ldots, 7$.

3.2 Determination of $\{s_i\}$ using $\hat{u}$

We use a polynomial approximation between the presumably smooth and edge-contained parts of the image similarly to the fully adaptive approach in [10]. Hence, let $0 < d \leq m < \infty$ be given numbers and $p_1(x)$ the second-order polynomial satisfying the conditions $p_1(d) = 1.5$, $p_1(m) = 1$, $p_1'(m) = 0$. Define the function

\[ p(x) = \begin{cases} 2, & 0 \leq x < d \\ p_1(x), & d \leq x \leq m \\ 1, & x > m \end{cases} \]

Then, using the defined function $p(x)$, we set

\[ s_i = p(|(D_i u)|), \quad i = 1, \ldots, n_1 n_2 - 1 \]

to be used in (6).

The reasons behind these choices are as follows: from [24], we know that having $s = 1.5$ in (3) is large enough for recovering smooth surfaces, and larger values of $s$ only oversmooth the images. However, to make problem (6) smooth enough for the CG method as a solver, we choose $s = 2$ near 0. The second number $m$ indicates the start of large enough gradients, i.e. edges. Finally, by choosing $p_1'(m) = 0$ we favour the non-smooth regularisation $s = 1$ in order to decrease the significance of the choice of $m$ and also to maintain the restoration of small features in an image. An example of interpolation function $p$ is given in Fig. 2.

Finally, we need to choose the values of $d$ and $m$. Let $H_{ab}(u)$ denote the histogram of the data $D(u) = \{(D_i u)_i, i = 1, \ldots, n_1 n_2 - 1\}$, binned to $nb$ bins. We use histograms $H_{50}(u_{TV})$ for choosing the values of $d$ and $m$. First, $d$ is set to be the width of the first bin of the histogram. In order to choose the value of $m$, we compute the derivative of the function described by (the shape of) the histogram $H_{50}(u_{TV})$. Then, we set $m$ to be the point at which the derivative of the histogram starts to grow for the first time. Two examples of such histograms are shown in Fig. 11, where the shapes of the derivatives are also plotted. The choice of $m$ is illustrated by a vertical line in Fig. 11.

Remark 3.1: It must be noted that this is only an example of the choice of the interpolation function $p$ and its parametrisation. For different application domains, the choice of $p$ and especially of $m$ must be reconsidered (cf. [27]).

3.3 Solution of problem (6)

Problem (6) is solved using the conjugate gradient method of Polak-Ribiére with inexact line search using the strong Wolfe conditions [29]. We choose $u^0 = u_{TV}$ and as a stopping criterion $\|u^k - u^{k-1}\|_\infty < 10^{-5}$.

4 Numerical experiments

Here, we present the results of numerical experiments obtained using the SAC method described in the previous section. The purposes of the experiments were to study the restoration capability and efficiency by means of the elapsed CPU time of the SAC method. Another purpose of this section is to illustrate the performance of each step of the SAC method.

![Fig. 1 Selection of g](image1)

![Fig. 2 Example of function p](image2)
The computations were performed on an HP9000/35600, 552 MHz, PA8600 CPU. The tested algorithms were implemented using Fortran 77. Computations were based on double-precision arithmetics, and the grey-level images were produced for illustrative purposes. However, since the grey-level image is not very useful in comparing the quality of images, we mainly use intensity plots and plots of the gradient for qualitative comparison.

4.1 Example images
The following example images were used in the numerical experiments. Noisy images \( z \) were formed by adding normally distributed random noise from \( \mathcal{N}(0,1) \), scaled with the noise level \( \eta \), to the original double-precision image \( z^* \).

**Example 4.1:** The original image \( z^* \) presents a self-generated double-precision image of size 300 \( \times \) 300, with double-precision values on the interval \([0,2]\). Original and noisy grey-level images are given in Fig. 3, where the transformation into grey-level images is based on 8-bit quantisation, so that the grey-level values vary on the interval \([0,255]\). In Fig. 5, a part of the image is presented as an intensity plot to illustrate the amount of noise contained in the image. The noise level is \( \eta = 0.3 \).

**Example 4.2:** The original 401 \( \times \) 381 true-colour RGB image (3 \( \times \) 8-bit quantised) was taken by a Nokia 7650 mobile phone. The grey-level image in this example consists of the red colour component, where the 8-bit integer values were scaled on to the interval \([0,1]\) to obtain the double-precision image \( z^* \). Original and noisy grey-level images are given in Fig. 4. In Fig. 6, the intensity plots present the lower right quadrant of the image. The noise level is \( \eta = 0.1 \).

![Fig. 3](image1.png) *Grey-level images in Example 4.1 (true, noisy, SAC, respectively)*

![Fig. 4](image2.png) *Grey-level images in Example 4.2 (true, noisy, SAC, respectively)*

![Fig. 5](image3.png) *True and noisy images (a, b) and the BV (c) and SAC (d) results in Example 4.1*
Example 4.3: The original image presents a grey-level image of the spine, taken from Matlab. The size of the image is $490 \times 367$, and originally it has been presented using only 6 bits, so that the grey-level values of the image vary on the interval $[0, 64]$. For our computations, the original image was scaled onto the interval $[0, 1]$, and this scaled double-precision image is denoted by $z'$. In Fig. 7, the intensity plots again present a part of the original and noisy images. The noise level is $\eta = 0.1$.

The gradients of true and noisy images in Examples 4.1–4.2 are plotted in Figs. 8 and 9. Different segments of the true images are clearly visible on the left, whereas the plots for the noisy images are absolutely non-informative.

4.2 Results obtained

In this Section, we illustrate the performance of each step of the SAC method by using the example images given in Section 4.1. In the experiments, we chose $\mu = 10^{-10}$ in (3) and (6).
4.2.1 Determination of \( g \): In Fig. 10, the results of Algorithm 3.1 are illustrated for \( G = \{8 \times 10^{-3}, 9 \times 10^{-3}, 1 \times 10^{-4}, 2 \times 10^{-4}, \ldots, 9 \times 10^{-4}, 1 \times 10^{-3}, 2 \times 10^{-3}, \ldots, 6 \times 10^{-3}\} \). In the lower plot, the value of the regularisation part \( r(g) \) is plotted for the seven values of \( g \). In the upper plot, the RMS reconstruction error \( e(\mu^*) \) is plotted and the value of \( g \) giving the smallest value of \( r(g) \) is marked with a star. The results show that the value of \( g \) yielding the smallest value of \( r(g) \) gives a good approximation of the best \( g \) with respect to \( e(\mu^*) \). The choices of \( g \) we obtained for Examples 4.1–4.3 were \( g = 10^{-3}, 3 \times 10^{-4}, 3 \times 10^{-4}, \) respectively.

4.2.2 Reference solutions: Plots of the TV results are given in Figs. 5–7 and plots of the gradients of the TV results for Examples 4.1–4.2 are given in Figs. 8 and 9. Our first conclusion is that the more noise the image contains, the more visible is the staircase-like structure in the TV result, due to the choice of larger \( g \). Hence, we summarise that when we use the proposed choice of \( g \) (which is probably smaller than in most other work [10, 12, 22, 28]), the TV result already yields a significant reduction in noise and improvement in image quality.
4.2.3 Determination of $s_i$: For Examples 4.1 and 4.2, the histograms $H_{s_0}(u_{TV})$ are shown in Fig. 11. Hence, $d$ is set to be the width of the first bin of the histogram and the choice of $m$ is illustrated by a vertical line in Fig. 11.

4.2.4 SAC Results: Plots of the SAC results are given in Figs. 5–7. For the SAC results of Examples 4.1 and 4.2, grey-level images are shown in Figs. 3 and 4 and plots of the gradients in Figs. 8 and 9. Finally, plots of the gradients of the BV and SAC results for Example 4.3 are given in Fig. 12. Both intensity and gradient plots show that the SAC result is clearly smoother than the TV result, while the jumps (edges) are still well preserved.

4.2.5 Results tables: In Table 1, $e(z)$ denotes the RMS reconstruction error of the noisy image, and $e(u_{TV})$, $e(u_{SAC})$ the errors of the TV and SAC results, respectively. The error $e(u_{SAC})$ is always smaller than $e(u_{TV})$, in Examples 4.1 and 4.2 the difference is considerable.

In Table 2, the elapsed CPU times are given for Step 1 (i.e. solving the TV problem seven times) and for Step 3 (i.e. solving the SAC problem 6) using CG of the SAC algorithm. The CPU times vary significantly between examples. For Examples 4.1 and 4.2, solving the SAC problem takes longer than Step 1 of the algorithm, whereas for Example 4.3, Step 1 of the algorithm takes three times longer than the solution of the SAC problem.

In Table 3, the error $e(u_{SAC})$, and the elapsed CPU time (determining $g$ + solving the SAC problem) are given for different values of the noise level $\eta$ in Example 4.1. A similar table was presented in [24] for a non-adaptive ($\delta$-) formulation. The example was the same but the noise was generated using a different random number generator. In all cases, we conclude that the SAC method gives smaller errors for all values of $\eta$ than any of the methods considered in [24].

4.2.6 Conclusions from the results: When we used the values of $g$ given by the heuristic method, the TV result yielded a significant reduction in noise and improvement in image quality. Moreover, the SAC result was clearly smoother than the TV result, while the jumps (edges) were still well preserved. Also the RMS reconstruction error indicated that the SAC result was better than the TV result.

The CPU times varied significantly between the examples. Anyway, times were reasonable, when we take into account that the regularisation parameter was determined during the same procedure. It must be noted that the SAC problem (6) was solved using the standard CG. To solve the problem, more efficient methods can be certainly be developed.

5 Conclusions

An optimisation methodology based on a semi-adaptive, convex formulation for the image denoising problem was proposed. The SAC formulation is always strictly convex and smooth enough that well-known gradient-based solution methods can be applied as part of the solution process. Another principal part of the algorithm relies on the fast active-set method for solving the TV-regularised image restoration problem. Using the heuristic determination of the regularisation parameter, no a priori information is needed concerning the amount of noise contained in the given original image.

The SAC method presented in this paper performs post-processing of the TV result that reduces the staircase effect while preserving the sharp edges. This was verified in numerical experiments. In the experiments presented the SAC results were better than the TV results, both quantitatively (RMS error) and qualitatively (plots).

Finally, we describe some modifications that can be useful when applying the SAC method to real applications.

5.1 Iterating SAC

Naturally, the SAC method can be iterated. This means that the current SAC result $u_{SAC}$ is used as a reference solution $\bar{u}$ for calculating a new $\{s_i\}$ for problem (6). Notice that such an iterative procedure can be considered a special kind of linearisation scheme for the fully adaptive approach in [10]. In our examples, iterating SAC two or three times made the results worse by oversmoothing them (cf. sweeping the median filter).

5.2 Search interval $G$

The search interval $G = [8 \times 10^{-5}, 6 \times 10^{-3}]$ that we used for $g$ may not always be the best one. By looking at the plot of $r(u)$, it is easy to tell if the search interval has the right location: $r(u)$ is convex in a neighbourhood of the best $g$. In practice, we may also decrease the size of the initial search interval (and vector $G$) to speed up the search.

Table 1: Errors for the results

<table>
<thead>
<tr>
<th>Ex</th>
<th>$e(z)$</th>
<th>$e(u_{TV})$</th>
<th>$e(u_{SAC})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>0.298</td>
<td>$3.34 \cdot 10^{-2}$</td>
<td>$2.68 \cdot 10^{-2}$</td>
</tr>
<tr>
<td>4.2</td>
<td>0.100</td>
<td>$2.98 \cdot 10^{-2}$</td>
<td>$1.85 \cdot 10^{-2}$</td>
</tr>
<tr>
<td>4.3</td>
<td>0.100</td>
<td>$2.11 \cdot 10^{-2}$</td>
<td>$2.01 \cdot 10^{-2}$</td>
</tr>
</tbody>
</table>

Table 2: CPU times for Steps 1 and 2 of the SAC algorithm

<table>
<thead>
<tr>
<th>Ex</th>
<th>Step 1</th>
<th>Step 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>99.85</td>
<td>148.94</td>
</tr>
<tr>
<td>4.2</td>
<td>31.99</td>
<td>101.56</td>
</tr>
<tr>
<td>4.3</td>
<td>218.99</td>
<td>69.16</td>
</tr>
</tbody>
</table>

Table 3: Results for different noise levels in Example 4.1

<table>
<thead>
<tr>
<th>$\eta$</th>
<th>$e(u_{SAC})$</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0.0068</td>
<td>71.26</td>
</tr>
<tr>
<td>0.1</td>
<td>0.0120</td>
<td>82.14</td>
</tr>
<tr>
<td>0.3</td>
<td>0.0268</td>
<td>248.79</td>
</tr>
<tr>
<td>0.5</td>
<td>0.0414</td>
<td>191.32</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0660</td>
<td>227.41</td>
</tr>
</tbody>
</table>

Fig. 12 Gradient images of the BV (a) and SAC (b) results for Example 4.3


559
5.3 Reference solution $\bar{u}$

Of course, the reference solution $\bar{u}$ can be other than $u_{TV}^*$, such as in the iterated SAC. We use $u_{TV}^*$ because of its known characteristic features, the fast AS algorithm for solving the TV problem, and the possibility to fix $g$ during the same phase.

5.4 Tuning the SAC method for a specific application area

To this end, the presented tests and our experience with the SAC method suggest that the method should be tuned for a specific application, for example, through the following steps:

1. Collect a representative set of sample images from the application domain
2. Compute the value of the regularisation parameter $g$ over the set of samples. If $g$ is about the same size for all the samples, it can be fixed for the unknown images to be restored.
3. In various application domains, images with different characteristics are encountered so that the determination of the interpolation function $p$ must be tuned individually for each application domain. For this purpose, one can utilise suitable histogram capturing the gradient information of the TV result.
4. Carry out tests concerning stopping criteria and the number of SAC iterations.

6 Acknowledgments

This work was supported by the Academy of Finland under FibNet and ProBisi projects.

7 References