



Heriot-Watt University  
Research Gateway

## Two Iterative Methods for Reverse Image Filtering

**Citation for published version:**

Belyaev, A & Fayolle, P-A 2021, 'Two Iterative Methods for Reverse Image Filtering', *Signal, Image and Video Processing*, vol. 15, no. 7, pp. 1565-1573. <https://doi.org/10.1007/s11760-021-01889-3>

**Digital Object Identifier (DOI):**

[10.1007/s11760-021-01889-3](https://doi.org/10.1007/s11760-021-01889-3)

**Link:**

[Link to publication record in Heriot-Watt Research Portal](#)

**Document Version:**

Publisher's PDF, also known as Version of record

**Published In:**

Signal, Image and Video Processing

**Publisher Rights Statement:**

© The Author(s) 2021.

**General rights**

Copyright for the publications made accessible via Heriot-Watt Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

**Take down policy**

Heriot-Watt University has made every reasonable effort to ensure that the content in Heriot-Watt Research Portal complies with UK legislation. If you believe that the public display of this file breaches copyright please contact [open.access@hw.ac.uk](mailto:open.access@hw.ac.uk) providing details, and we will remove access to the work immediately and investigate your claim.



# Two iterative methods for reverse image filtering

Alexander G. Belyaev<sup>1</sup> · Pierre-Alain Fayolle<sup>2</sup>

Received: 4 September 2020 / Revised: 7 February 2021 / Accepted: 4 March 2021  
© The Author(s) 2021

## Abstract

We consider the problem of recovering an original image  $\mathbf{x}$  from its filtered version  $\mathbf{y} = \mathbf{f}(\mathbf{x})$ , assuming that the internal structure of the filter  $\mathbf{f}(\cdot)$  is unknown to us (i.e., we can only query the filter as a black-box and, for example, cannot invert it). We present two new iterative methods to attack the problem, analyze, and evaluate them on various smoothing and edge-preserving image filters.

**Keywords** Reverse filtering · Nonlinear filtering · Recovery

## 1 Introduction and contribution

In this paper, we deal with a reverse imaging problem introduced recently in [19]. Given an image filter  $\mathbf{y} = \mathbf{f}(\mathbf{x})$  with input  $\mathbf{x} \in \mathbb{R}^d$  and output  $\mathbf{y} \in \mathbb{R}^d$ , the problem consists of restoration of  $\mathbf{x}$  from  $\mathbf{y}$  under the condition that we can apply the filter  $\mathbf{f}(\cdot)$  as many times as needed but without any knowledge of its internal structure. In particular, it means that we cannot directly invert the filter. Following [19], we call such a restoration process *reverse filtering* or *defiltering*.

The reverse image filtering schemes introduced in [19] and in the subsequent works [4,12] are aimed at removing slight filtering effects. In contrast, our work is focused on removing, at least partially, severe filtering effects. Three examples are shown in Fig. 1, where reverse filtering of images filtered by a Gaussian, Laplace-of-Gaussian (LoG), and the adaptive manifold filter (AMF) [6] is demonstrated.

Mathematically, the defiltering problem consists of numerically solving the equation

$$\mathbf{y} = \mathbf{f}(\mathbf{x}), \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^d, \quad (1)$$

for  $\mathbf{x}$ , given  $\mathbf{y}$ . Here,  $\mathbf{x}$  is an (unknown) input image,  $\mathbf{f}(\cdot)$  is a (black-box) filter, and  $\mathbf{y}$  is the (given) output image.

Numerical methods for solving (1) include the Newton's method and its relatives, various gradient descent schemes, and fixed-point iterations. Straightforward uses of the Newton's and gradient descent methods require an estimation of the Jacobian of  $\mathbf{f}(\cdot)$  and are not computationally feasible, as a typical image consists of tens of thousands to millions of pixels.

A simple and efficient fixed-point iteration scheme for image defiltering was recently proposed by Tao et al. [19]. The scheme consists of the following iterations

$$\mathbf{x}_{n+1} = \mathbf{T}(\mathbf{x}_n) \equiv \mathbf{x}_n + \mathbf{h}_n, \quad \mathbf{h}_n = \mathbf{y} - \mathbf{f}(\mathbf{x}_n), \quad \mathbf{x}_0 = \mathbf{y} \quad (2)$$

and can be considered as an iterative unsharp masking with filter  $\mathbf{f}(\cdot)$ . Mathematically, (2) corresponds to the Picard's iterative method  $\mathbf{x}_{n+1} = \mathbf{f}(\mathbf{x}_n)$ , a simple and popular method for numerically solving nonlinear equation (1) (see, for example, [1] for a comprehensive study of fixed-point iteration methods). A rigorous convergence analysis of (2) for image filters satisfying certain conditions is given in [19].

As shown in [19], (2) demonstrates excellent results for removing light filtering effects but, according to our experiments, often fails when dealing with severe image deformations. So in this paper, we introduce two iterative methods

$$\mathbf{x}_{n+1} = \mathbf{P}(\mathbf{x}_n) \equiv \mathbf{x}_n + \lambda_n \frac{\mathbf{f}(\mathbf{x}_n + \mathbf{h}_n) - \mathbf{f}(\mathbf{x}_n - \mathbf{h}_n)}{2},$$
$$\lambda_n = \frac{4\|\mathbf{h}_n\|^2}{\|\mathbf{f}(\mathbf{x}_n + \mathbf{h}_n) - \mathbf{f}(\mathbf{x}_n - \mathbf{h}_n)\|^2},$$

✉ Alexander G. Belyaev  
a.belyaev@hw.ac.uk

Pierre-Alain Fayolle  
fayolle@u-aizu.ac.jp

<sup>1</sup> School of Engineering and Physical Sciences, Heriot-Watt University, Edinburgh, UK

<sup>2</sup> Computer Graphics Laboratory, University of Aizu, Aizu-Wakamatsu, Japan



**Fig. 1** Top row: test images we deal with in this paper: Trui, cameraman, and Barbara. Middle row: the images are filtered by a Gaussian `imgaussfilt(x,5)`, Laplace-of-Gaussian `imfilter(x,fspecial('log',7,0.4))`, and the adaptive manifold filter (AMF) `adaptive_manifold_filter(x,20,0.4)` [6], respectively (Matlab notations are used to specify the parameters of the filters). Bottom row: defiltering results; notice how well the fine image details are restored

$$h_n = y - f(x_n), \quad x_0 = y \tag{3}$$

and

$$x_{n+1} = S(x_n) \equiv x_n + \frac{h_n \|h_n\|}{\|f(x_n + h_n) - f(x_n)\|}, \quad x_0 = y \tag{4}$$

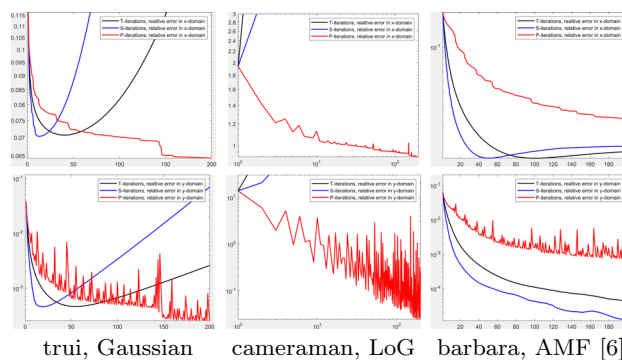
which are inspired by a gradient descent scheme studied by Polyak [15] and Steffensen’s method [17], respectively (hence the notations  $P(\cdot)$  and  $S(\cdot)$ ). We propose a brief analysis of (2), (3) and (4) in the next section.

Given  $x_n$  the reconstructed image at the  $n$ -th step of the iterative process, we consider two error functions in order to measure the reconstruction accuracy

$$E_x(x_n) = \frac{\|x - x_n\|}{\|x\|} \quad \text{and} \quad E_y(x_n) = \frac{\|y - f(x_n)\|}{\|y\|}. \tag{5}$$

Although we use both error functions to evaluate the T, P, and S-iterative processes, only  $E_y$  can be used in practice, as  $x$  is not known a priori.

Figure 2 presents the  $E_x$  and  $E_y$  error graphs corresponding to the reverse filtering examples shown in Fig. 1.



**Fig. 2** The error graphs for  $E_x$  (top row) and  $E_y$  (bottom row) for the reverse filtering examples of Fig. 1. Results for the first 200 iterations are presented (additional iterations are needed for defiltering the Gaussian and LoG filtering effects). For LoG, middle column, log–log plots are used to keep the results of the T and S-iterations visible in the graph

## 2 Analysis of the previous and proposed methods

Numerical methods for solving (1) include Newton’s method and its relatives, as well as various gradient descent schemes, and fixed-point iterations [13]. Unfortunately, the vast majority of these methods require knowledge of the gradient of  $f(\cdot)$  or impose too many restrictions on the behavior of  $f(\cdot)$ .

Tao et al. [19] use a fixed-point iteration method to analyze convergence properties of their T-iterations (2). Unfortunately, their analysis can be applied only to linear mappings.

A similar scheme

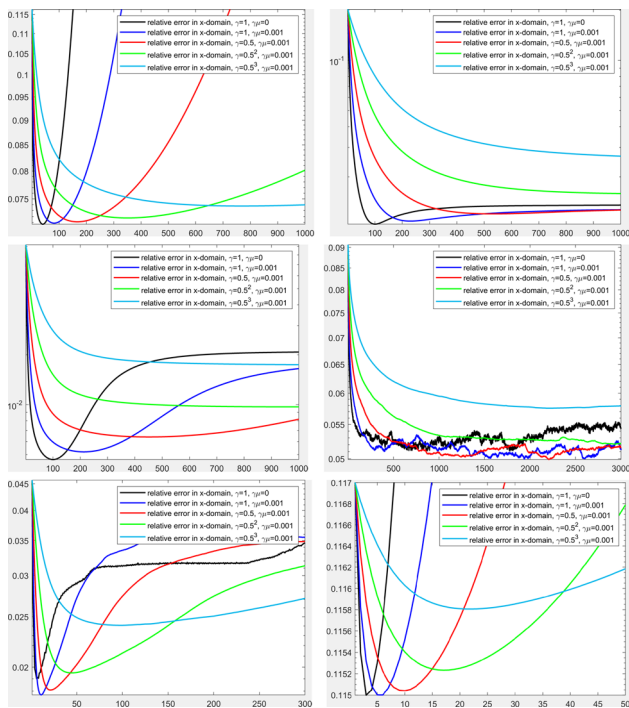
$$x_{n+1} = M(x_n) \equiv (1 - \gamma\mu)x_n + \gamma(y - f(x_n)), \quad x_0 = y, \tag{6}$$

was proposed by Milanfar [12] who used different considerations. Here,  $\gamma$  and  $\mu$  are positive parameters chosen such that both  $\gamma < 1$  and  $\gamma\mu$  is small. Although the iterative scheme (6) is supported by a solid analysis of its convergence properties [12], our experiments have not revealed practical advantages of (6) over (2). In Fig. 3 we present the  $E_x$ -error graphs for defiltering six linear and nonlinear filters using (2) and (6) with  $\gamma\mu = 0.001$  and  $\mu = 0.5^k$ ,  $k = 0, 1, 2, 3$ . One can see that in the majority of cases, (2) slightly outperforms (6).

In [4], an interesting iterative scheme for solving (1) was proposed

$$x_{n+1} = \mathcal{F}^{-1} \left( \frac{\mathcal{F}(y)\mathcal{F}(x_n)}{\mathcal{F}(f(x_n))} \right). \tag{7}$$

Here,  $\mathcal{F}$  stands for the Fourier transform and element-wise multiplication and division are used. Interestingly, (7) can be considered as a variant of (6) employed in the frequency domain and used with variable parameter  $\gamma_n$  instead of  $\gamma$  and



**Fig. 3**  $E_x$ -error graphs for defiltering (2) and (6). Top-left: Gaussian; top-right: AMF [6]; middle-left: bilateral filtering; middle-right: RTV [24]; bottom-left: WMF [26]; bottom-right: RGF [25]. In the majority of these tests (2), corresponding to the curves in black, slightly outperforms (6)

$\mu = 0$ . Indeed, the iterative process

$$\mathcal{F}(x_{n+1}) = \mathcal{F}(x_n) + \gamma_n (\mathcal{F}(y) - \mathcal{F}(f(x_n)))$$

with  $\gamma_n = \mathcal{F}(x_n) / \mathcal{F}(f(x_n))$

yields (7). This iterative scheme shows an excellent performance for linear filters  $f(\cdot)$ , provided that periodic boundary conditions are imposed, as demonstrated by the top-left and top-middle images of Fig. 4. Unfortunately, if other types of boundary conditions are chosen, (7) does not lead to good defiltering results, as shown by the bottom-left and bottom-middle images. In addition, this method does not show a good performance for some nonlinear filters, as seen in the right images of Fig. 4.

It is interesting to note that the problem of reverse image filtering is closely related to stochastic approximation [2,9], an active research area initiated by Robbins and Monro (RM) almost seventy years ago [16]. Remarkably, the RM algorithm looks very similar to (2). Given noisy measurements

$$y_n = f(x_n) + \eta_n,$$

where  $\{\eta_n\}$  are measurement errors, the algorithm implements a simple iterative procedure for solving the equation



**Fig. 4** Defiltering `imgaussfilt(x, 5, 'Padding', 'circular')` (top-left) and `imgaussfilt(x, 5)` (bottom-left) by (7). Similar defiltering results for `imfilter(x, fspecial('log', 7, 0.4))` with (top-middle) and without (bottom-middle) periodic boundary conditions by (7). Top-right: defiltering RTV [24] with default settings by (7). Bottom-right: defiltering AMF [6] `adaptive_manifold_filter(x, 20, 0.4)` by (7)

$$y = f(x)$$

$$x_{n+1} = x_n + a_n(y - y_n),$$

where  $\{a_n\}$  is a sequence of nonnegative real numbers satisfying  $\sum a_n = \infty$  and  $\sum a_n^2 < \infty$ .

Our P-iterations (3) can be considered as an approximation of the gradient descent with a variable step-size for minimizing least-square energy

$$\mathcal{E}(x) = \|y - f(x)\|^2 \tag{8}$$

whose anti-gradient is approximated by a symmetric finite difference

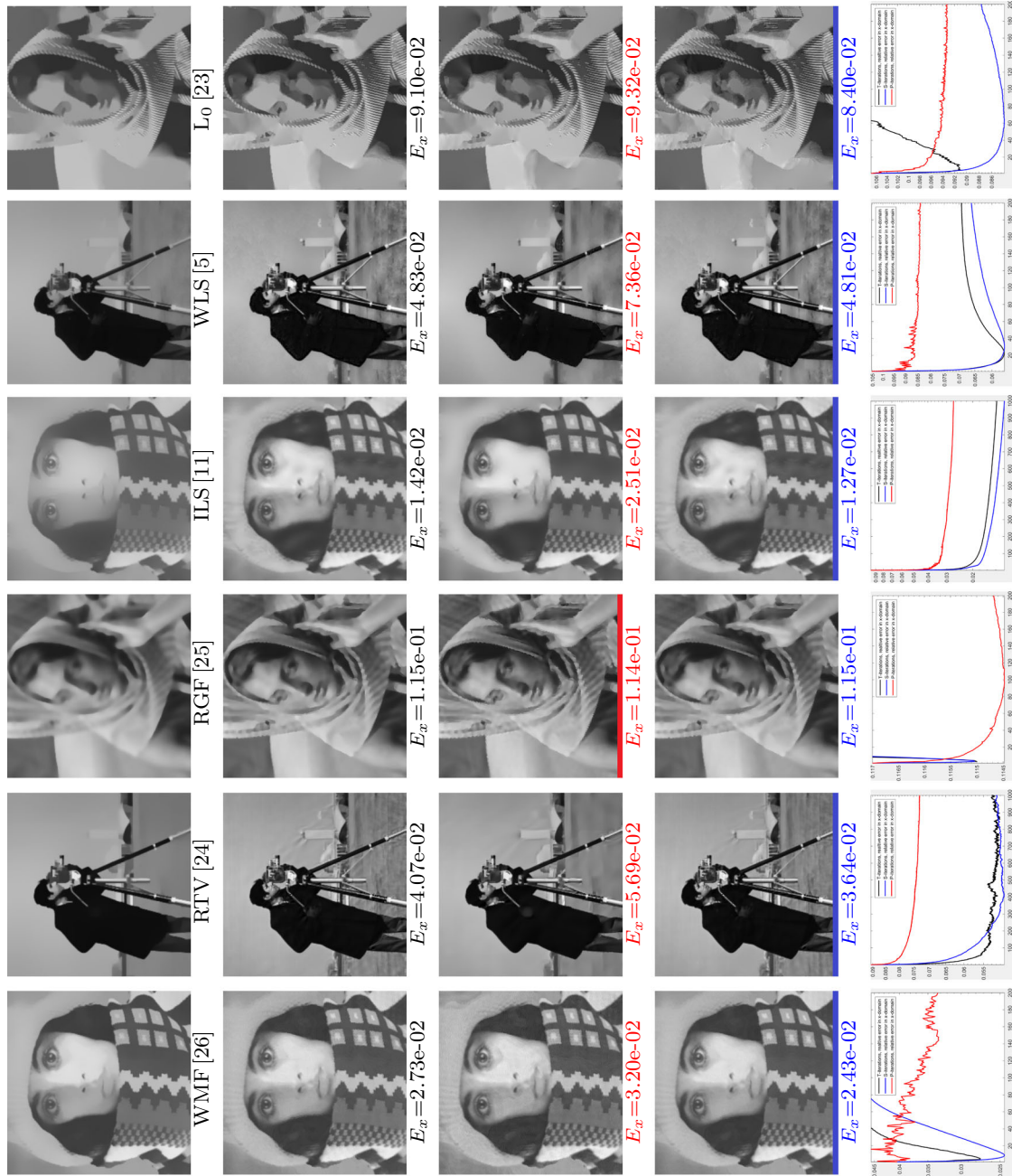
$$-\nabla \mathcal{E}(x_n) = 2\nabla f(x_n)h_n \approx f(x_n + h_n) - f(x_n - h_n),$$

where, as before,  $h_n = y - f(x_n)$  is assumed to be sufficiently small.

It turns out that in the one-dimensional case (single-pixel images), our iterative scheme (3) approximates the standard Newton iterations for numerically solving  $y = f(x)$ . Indeed, the Newton iterations for solving  $F(x) \equiv y - f(x) = 0$  are given by

$$x_{n+1} = x_n - \frac{F(x_n)}{F'(x_n)} = x_n + \frac{h_n}{f'(x_n)}, \quad h_n = y - f(x_n).$$





**Fig. 5** Results obtained on several state-of-the-art edge-preserving filters. From top to bottom: the images are defiltered by T-iterations (2); third row: the images are defiltered by P-iterations (3); fourth row: the images are defiltered by S-iterations (4); fifth row: the corresponding  $E_x$ -error graphs. For each image, its  $E_x$ -error value is shown. For each filtering method, the image with the minimum  $E_x$ -error value is marked by a thick color line, where black corresponds to the T-iterations (2), red to the P-iterations (3), and blue to the S-iterations (4)

Now we can observe that in the one-dimensional case (3) is reduced to

$$x_{n+1} = x_n + \frac{2h_n^2}{f(x_n + h_n) - f(x_n - h_n)} \approx x_n + \frac{h_n}{f'(x_n)}.$$

It is also worth mentioning a link between (3) and the Kiefer–Wolfowitz (KW) stochastic approximation method [8]. The KW algorithm for a univariate function  $y = f(x)$  consists of an iterative process

$$x_{n+1} = x_n + a_n \frac{f(x_n + c_n) - f(x_n - c_n)}{2c_n}, \quad (9)$$

where  $c_n \rightarrow 0$  in addition to some other conditions imposed on the sequences  $\{a_n\}$  and  $\{c_n\}$ . A multivariate version of the KW algorithm can be found, for example, in [9] (initially both the RM and KW algorithms were proposed by their authors for univariate functions).

In the one-dimensional case, our S-iterations (4) become Steffensen's method [17]

$$x_{n+1} = x_n + \frac{h_n}{f(x_n + h_n) - f(x_n)} h_n, \quad h_n = y - f(x_n),$$

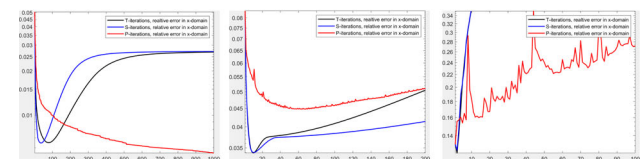
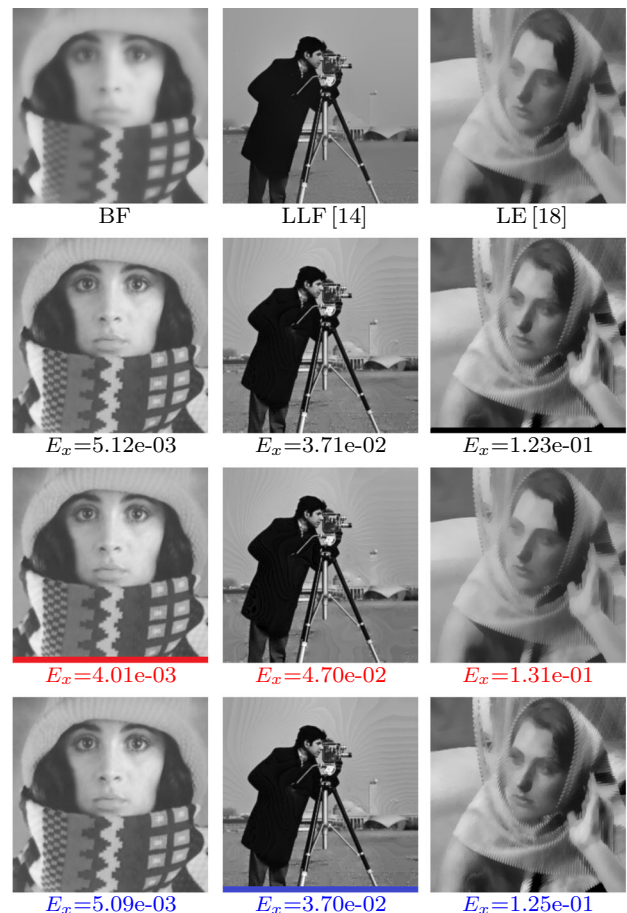
which delivers a numerical solution to  $y = f(x)$  and has quadratic convergence like Newton's method.

### 3 Numerical experiments and discussion

In this section, we test our methods (3) and (4), and compare them against (2) using a wide selection of edge-preserving image filters, in addition to the two linear filters (Gaussian and LoG) and the adapted manifold filter (AMF) [6] considered in Sect. 1.

Results obtained are shown in Fig. 5 for the image filters: WMF [26], RTV [24], RGF [25], ILS [11], WLS [5] and  $L_0$  [23], and in Fig. 6 for the bilateral filter (BF), LLF [14] and LE [18]. For each image filter, we show an example of a filtered test image (top row), followed by the results obtained from T-iterations of Tao et al. [19] (2), P-iterations (3) and S-iterations (4) in the second, third and fourth row (from the top). For each method, the minimal  $E_x$ -error is indicated under the image. The bottom row provides error graphs for  $E_x$  (5) for the three methods. The best method is decided based on the error function  $E_x$  (5) and indicated by a thick color line. Table 1 summarizes the best methods for each filter according to the  $E_x$ -error function.

For the filters used in our experiments and shown in Figs. 5 and 6, we use the Matlab implementations provided by the authors of the corresponding papers with the following settings (we use Matlab notations for the filters and their arguments):



**Fig. 6** Further experiments with the state-of-the-art edge-preserving image filters. From top to bottom: first row: filtered images; second row: the images are defiltered by T-iterations (2); third row: the images are defiltered by P-iterations (3); fourth row: the images are defiltered by S-iterations (4); fifth row: the corresponding  $E_x$ -error graphs. For each image, its  $E_x$ -error value is presented. For each filtering method, the image with the minimum  $E_x$ -error value is marked by a thick color line, where black corresponds to the T-iterations (2), red to the P-iterations (3), and blue to the S-iterations (4)

```

WMF[26]: jointWMF(x),
RTV[24]: tsmooth(x),
RGF[25]: RollingGuidanceFilter(x, 3, 0.05, 4),
ILS[11]: ILS_LNorm(x, 1, 0.8, 0.0001, 4),
WLS[5]: wlsFilter(x),
L0[23]: L0Smoothing(x, 0.01),
BF: imbilatfilt(x, 0.05, 3),
LLF[14]: double(locallapfilt(im2uint8
(x), 0.2, 10.0))/255,
LE[18]: localExtrema(x, x, 7).

```

**Table 1** For each filter, the best defiltering method is indicated

Gaussian & LoG	RTV [24]	LE [18]	BF	RGF [25]	WLS [5]	WMF [26]	L <sub>0</sub> [23]	AMF [6]	ILS [11]	LLF [14]
P	S	T, S	P	P	T, S	S	S	T, S	S	T, S

The decision is made based on the error function  $E_x$ . P stands for P-iterations (3), S for S-iterations (4), and T for T-iterations (2)

Surprisingly, reverse filtering of linear filters (in this paper, we consider Gaussian and LoG filters) turns out to be a difficult task. In the case of periodic boundary conditions, perfect defiltering is delivered by the method of Dong et al. (7) proposed in [4]. However, (7) is very sensitive to small perturbations and fails to recover images resulting from linear filters with non-periodic boundary conditions (see, e.g., Fig. 4).

The method of Tao et al. (2) [19] works well when it is used for defiltering Gaussians with small variances. In our example of Gaussian defiltering (see the left images in Figs. 1 and 2), the variance is  $\sigma^2 = 5$  and (2) fails to produce a reasonably good result. Our experiments show that (2) also fails to recover the LoG filtering results (see, for example, Figs. 1 and 2, middle column).

Our P-iteration scheme (3) is capable of defiltering the Gaussian and LoG filters, as demonstrated in Figs. 1 and 2. However, the convergence can be slow. For example, restoring the left and middle images of Fig. 1 requires 100K iterations.

In addition to the Gaussian and LoG filters, we test ten state-of-the-art edge-preserving filters, see Figs. 5 and 6 and the right images of Figs. 1 and 2. For some filters, e.g., AMF, excellent defiltering results are obtained. For some others, e.g., LE, only very modest results are achieved. In general, we see that our P-iteration and S-iteration schemes together outperform the T-iteration method (2) [19]. However, for zero-order reverse filtering, (2) is the fastest method, as it requires only one call of function  $f(\cdot)$  per iteration and, if it approaches an error minimum value, does it quickly.

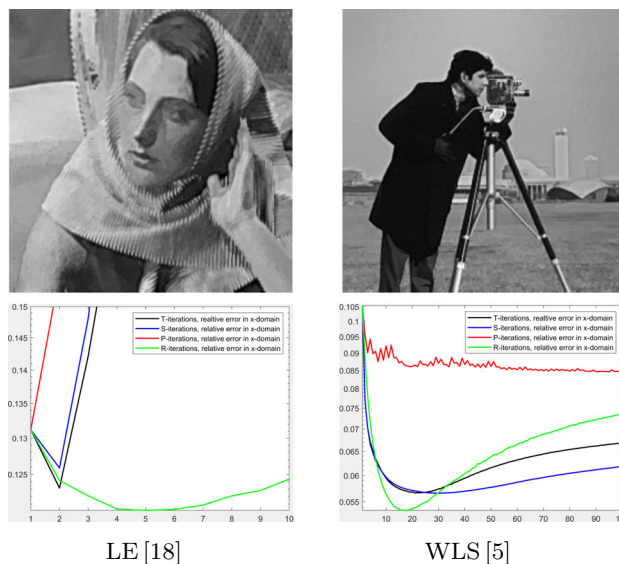
We find it surprising that for the majority of the tested filters, small-scale image details can be accurately recovered if a proper defiltering method is applied. This means that typically edge-preserving filters only suppress those small-scale details but do not remove them completely.

In our experiments, we observe only a weak correlation between the two error functions  $E_x$  and  $E_y$  (5) for an iteratively defiltered image  $x_n$ . Namely, if  $E_y$  does not grow, then it is very likely that the defiltered image has its main features preserved, as demonstrated by Fig. 7. In real-life scenarios when only  $E_y$  is available, the number of reverse filtering iterations has to be selected manually. This is the only user-specified parameter in our approach.

While, in this paper, we deal with deterministic reverse filtering methods, we feel that randomized algorithms could be very successful at achieving high-quality defiltering results.



**Fig. 7** Overrestoration examples. Left: WMF defiltering with 100 iterations of (4) amplifies certain middle-range frequencies. Right: BF defiltering with 1K iterations of (4) amplifies some high frequencies. Compare these results with those for WMF and BF from Figures 5 and 6, respectively



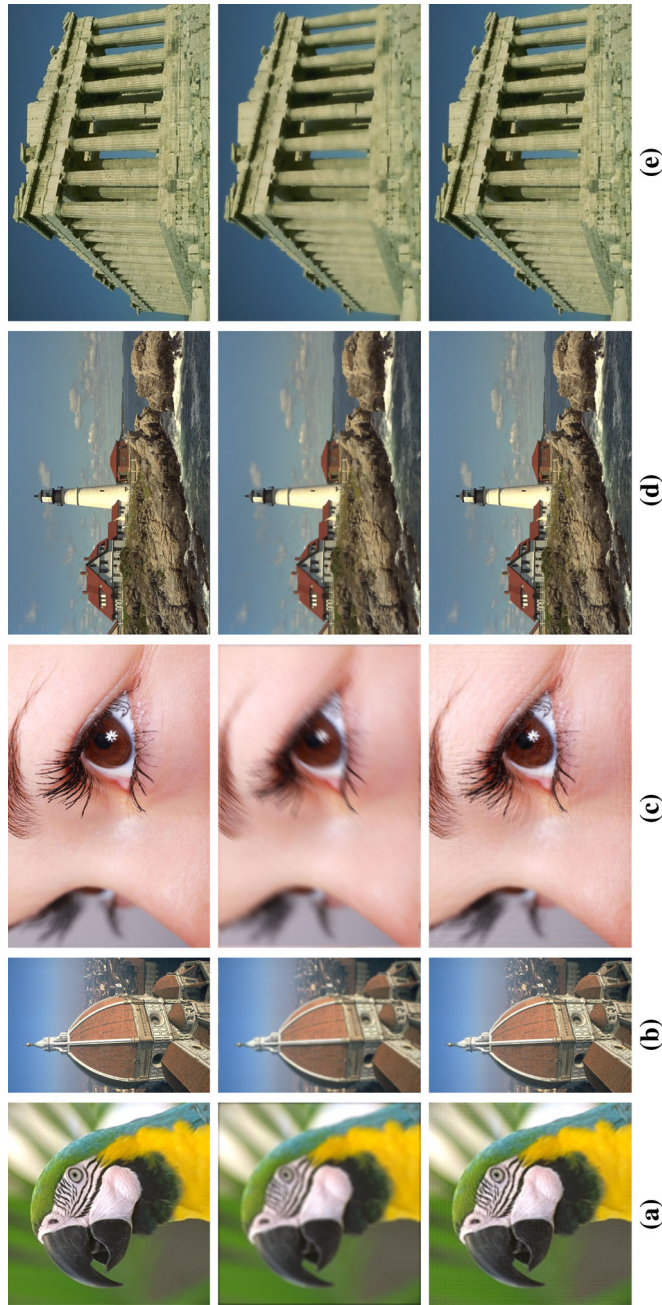
**Fig. 8** R-iterations (10) achieve the smallest  $E_x$ -errors for the LE filter [18] (left) and WLS filter [5] (right). Top: images defiltered by (10). Bottom: the corresponding  $E_x$ -error graphs

As an example, let us consider the following simple stochastic iterative process

$$x_{n+1} = R(x_n) \equiv x_n - \frac{\mathcal{E}(x_n + r \frac{c_n}{2}) - \mathcal{E}(x_n - r \frac{c_n}{2})}{c_n} \tag{10}$$

where quadratic energy  $\mathcal{E}$  is defined by (8),  $r$  is a random image whose components are random numbers uniformly distributed on the interval  $[0, 1]$  and  $c_n = 1/\sqrt{n}$ . One can see that (10) combines (9) with stochastic gradient descent ideas.





**Fig. 9** Top row: original images. Middle row: filtered images. Bottom row: the corresponding defiltered images. For each filtered image, the number of reverse filtering iterations is determined by (1) with  $\tau = 0.0005$ . **a** Wiener filtering and defiltering with (3). **b** Pillbox filtering and defiltering with (3). **c** Motion blur filtering and defiltering with (3). **d** Image-guided filtering and defiltering with (4). **e** Image-guided filtering using a smoothed guidance and defiltering with (4). See the main text for details. Zoom in the images to see how well small-scale details are restored



While (10) shows a modest performance for the majority of the considered filters, it allows us to achieve some progress with defiltering the LE filter [18] for which (10) shows the lowest  $E_x$  error. In Fig. 8, we show the result obtained by (10) for defiltering the LE and WLS filters. The top row shows the best obtained results, while the bottom row compares the result of the R-iterations (10) against the results of the T, S and P-iterations for the  $E_x$  error function. Visually (10) does a better work for retrieving fine details and high frequency texture in the image.

When dealing with color images, the simplest approach consists of defiltering each RGB channel. Obviously, more sophisticated strategies should lead to better results. Figure 9 shows several examples of color images, defiltered with our approach, from different domains (nature, architecture, humans, animals). In these examples, we test defiltering capabilities of our schemes (3) and (4) using popular image filters which have not yet been considered so far in this paper: (a) Wiener filter, (b) circular averaging filter (pillbox), (c) motion blur, (d) image guided filter [7], and (e) image guided filter, where Gaussian smoothing is applied to the guidance image:

- (a) `wiener2(x, [5, 5], 0.1)`
- (b) `imfilter(x, fspecial('disk', 3))`
- (c) `imfilter(x, fspecial('motion', 20, 45))`
- (d) `imguidedfilter(x)`
- (e) `imguidedfilter(x, imgaussfilt(x, 5))`

For the linear filters (first three filters), reverse filtering was done by applying (3). The remaining nonlinear filters were defiltered by (4). In these five examples, for each color channel, we stop iterative processes (3) and (4) immediately after the relative change in successive iterations becomes less than a user-specified threshold  $t > 0$

$$\|\mathbf{x}_{n+1} - \mathbf{x}_n\| / \|\mathbf{x}_n\| < t. \quad (11)$$

Better results could be obtained if an optimal number of iterations is chosen for each filter. Achieving reasonably good threshold  $t$  in (11) (for all the examples in Fig. 9 defiltering is done with  $t = 0.0005$ ) demonstrates the robustness of our schemes (3) and (4).

One seemingly promising application of defiltering schemes lies in image enhancing. It seems natural to use the proposed reverse image filtering schemes as follows: The image  $\mathbf{x}$  in (1) can be considered as an enhanced version of the input image  $\mathbf{y}$  if  $f(\cdot)$  is an appropriate filter. In Fig. 10, we combine AMF [6] with P-iterations (3) for image sharpening and with S-iterations (4) for image dehazing.



**Fig. 10** Original (left) and enhanced (right) images. Top: visually pleasant sharpening results can be achieved by combining AMF [6] and the P-iteration scheme (3) (in this example, ten iterations of (3) are used). Bottom: an image dehazing result is achieved by applying five S-iterations (4) based on the AMF filtering scheme

## 4 Conclusion and directions for future work

In this work, we have considered the problem of reverse filtering, or defiltering, a severely filtered image  $\mathbf{y} = f(\mathbf{x})$ . We assume no knowledge of the internal structure of the filter  $f(\cdot)$  (and thus cannot compute its inverse). Two proposed methods (3) and (4) are compared against the existing method (2) introduced in [19]. The comparison is done on severely filtered images.

Possible directions of future work on defiltering include the use of stochastic approximation methods for image defiltering and adapting image reverse filtering schemes for geometric modeling applications (see, for example, the recent work [22] devoted to reverse geometry filtering).

Using reverse filtering schemes for image enhancing and sharpening constitutes another direction for future work. See a very recent work [3] where single-step polynomial defiltering is used for fast and natural-looking sharpening large-size images.

Iterative schemes (3) and (4) can be considered as zero-order optimization methods. Problems requiring zero-order optimization appear frequently in signal processing and machine learning [10].

As (3) and (4) can be used to estimate how much of high-frequency image content is partially suppressed but remains preserved by an image filtering method, it would be interesting to establish a link with the ability of the method to protect deep learning schemes against adversarial attacks which often target high-frequency image components invisible to the human eye [20,21].

**Acknowledgements** We would like to thank the anonymous reviewers of this paper for their valuable and constructive comments.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

- Berinde, V.: Iterative approximation of fixed points. Lecture Notes in Mathematics, vol. 1912. Springer (2007)
- Chen, H.F.: Stochastic Approximation and Its Applications, Non-convex Optimization and Its Applications, vol. 64. Springer, New York (2006)
- Delbracio, M., Garcia-Dorado, I., Choi, S., Kelly, D., Milanfar, P.: Polyblur: Removing mild blur by polynomial reblurring. Tech. Rep. [arXiv:2012.09322](https://arxiv.org/abs/2012.09322) (2020)
- Dong, L., Zhou, J., Zou, C., Wang, Y.: Iterative first-order reverse image filtering. In: ACM Turing Celebration Conference-China, pp. 1–5 (2019)
- Farbman, Z., Fattal, R., Lischinski, D., Szeliski, R.: Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Trans. Graph.* **27**(3), 67:1–67:10 (2008)
- Gastal, E.S.L., Oliveira, M.M.: Adaptive manifolds for real-time high-dimensional filtering. *ACM Trans. Graph.* **31**(4), 33:1–33:13 (2012)
- He, K., Sun, J., Tang, X.: Guided image filtering. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(6), (2012)
- Kiefer, J., Wolfowitz, J.: Stochastic estimation of the maximum of a regression function. *Ann. Math. Stat.* **23**(3), 462–466 (1952)
- Kushner, H., Yin, G.G.: Stochastic Approximation and Recursive Algorithms and Applications, 2nd edn. Stochastic Modelling and Applied Probability. Springer, New York (2003)
- Liu, S., Chen, P.Y., Kailkhura, B., Zhang, G., Hero, A., Varshney, P.: A primer on zeroth-order optimization in signal processing and machine learning. *IEEE Signal Process. Mag.* (2020)
- Liu, W., Zhang, P., Huang, X., Yang, J., Shen, C., Reid, I.: Real-time image smoothing via iterative least squares. *ACM Trans. Graph.* **39**(3), 28:1–28:24 (2020)
- Milanfar, P.: Rendition: reclaiming what a black box takes away. *SIAM J. Imaging Sci.* **11**(4), 2722–2756 (2018)
- Ortega, J.M., Rheinboldt, W.C.: Iterative Solution of Nonlinear Equations in Several Variables. Academic Press, New York (1970)
- Paris, S., Hasinoff, S.W., Kautz, J.: Local laplacian filters: Edge-aware image processing with a Laplacian pyramid. *ACM Trans. Graph.* **30**(4), 68:1–68:11 (2011)
- Polyak, B.T.: Minimization of nonsmooth functionals. *U.S.S.R. Comput. Math. Math. Phys.* **9**(3), 14–29 (1969)
- Robbins, H., Monro, S.: A stochastic approximation method. *Ann. Math. Stat.* **22**(3), 400–407 (1951)
- Steffensen, J.: Remarks on iteration. *Skand. Aktuarietidskr.* **16**, 64–72 (1933)
- Subr, K., Soler, C., Durand, F.: Edge-preserving multiscale image decomposition based on local extrema. *ACM Trans. Graph.* **28**(5), 147:1–147:9 (2009)
- Tao, X., Zhou, C., Shen, X., Wang, J., Jia, J.: Zero-order reverse filtering. In: IEEE International Conference on Computer Vision (ICCV 2017), pp. 222–230. IEEE (2017)
- Wang, H., Wu, X., Huang, Z., Xing, E.P.: High-frequency component helps explain the generalization of convolutional neural networks. *Proceedings of CVPR 2020*, 8684–8694 (2020)
- Wang, Z., Yang, Y., Shrivastava, A., Rawal, V., Ding, Z.: Towards frequency-based explanation for robust CNN. Tech. Rep. [arXiv:2005.03141](https://arxiv.org/abs/2005.03141) (2020)
- Wei, M., Guo, X., Huang, J., Xie, H., Zong, H., Kwan, R., Wang, F.L., Qin, J.: Mesh defiltering via cascaded geometry recovery. *Comput. Graph. Forum* **38**(7), 591–605 (2019)
- Xu, L., Lu, C., Xu, Y., Jia, J.: Image smoothing via L0 gradient minimization. *ACM Trans. Graph.* **30**(6), 174:1–174:12 (2011)
- Xu, L., Yan, Q., Xia, Y., Jia, J.: Structure extraction from texture via relative total variation. *ACM Trans. Graph.* **31**(6), 139:1–139:10 (2012)
- Zhang, Q., Shen, X., Xu, L., Jia, J.: Rolling guidance filter. In: European Conference on Computer Vision (ECCV 2014), LNCS 8691, pp. 815–830 (2014)
- Zhang, Q., Xu, L., Jia, J.: 100+ times faster weighted median filter (WMF). In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2014), pp. 2830–2837 (2014)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.