An Efficient Algorithm for Computing the Derivative of Mean Structural Similarity Index Measure

Isabel Molina Orihuela[1,2] and Mehran Ebrahimi[2](✉)

[1] Universidad de Málaga, 31 Bulevar Louis Pasteur, 29010 Málaga, Spain
[2] University of Ontario Institute of Technology, 2000 Simcoe Street North, Oshawa, ON L1H 7K4, Canada
{isabel.orihuela,mehran.ebrahimi}@uoit.ca

Abstract. Many inverse problems in imaging can be addressed using energy minimization. The Euclidean distance is traditionally used in data fidelity term of energy functionals, even though it is not an optimal measure of visual quality. Recently the use of Mean Structural Similarity Index Measure (MSSIM) in data fidelity expressions has been examined. Solving such problems requires derivative of MSSIM. We propose an efficient algorithm for computing this derivative using convolutions. We indicate how the computational cost will be reduced to $\mathcal{O}(N \log N)$ from the cost of traditional scheme $\mathcal{O}(m\ N \log N)$ where N is the size of an input image and m is the window size. The proposed algorithm can be used for any inverse problem that traditionally applies L2 norm as a data fidelity measure. We apply the proposed numerical scheme to the inverse problem of image denoising.

Keywords: Structural Similarity Index Measure (SSIM) · Mean-SSIM · Inverse problems · SSIM derivative

1 Introduction

The analysis of image quality plays a central role in many image processing applications. Over the past decade, new objective image quality assessment measures have been developed that can better estimate image quality perceived by the human specialist [6]. Traditional measures such as the Peak Signal-Noise Ratio (PSNR) or Mean Square Error (MSE) are very simple to calculate, yet they do not necessarily relate to the image quality perceived by a human observer [6]. An original image can be altered with different distortions that are perceived differently, but all of them have the same MSE, see [6]. The proposed similarity measure in [6] known as Structural Similarity Index Measure (SSIM) is based on the hypothesis that the human visual system is adapted to extract structural information of images in such a way that a measure of the structural information can provide a good approximation of the perceived image quality.

© Springer Nature Switzerland AG 2019
F. Karray et al. (Eds.): ICIAR 2019, LNCS 11662, pp. 55–66, 2019.
https://doi.org/10.1007/978-3-030-27202-9_5

In the next Section, we first review the definitions of SSIM, MSSIM and the general formulation of inverse problems using MSSIM as a data fidelity term. This will be followed by a developing a novel technique for efficient derivative estimation of the MSSIM and proving in terms of computational complexity. Finally, preliminary experiments for total variation (TV)-based image denoising will be presented in which MSSIM is employed as a data fidelity term instead of the typical Euclidean distance.

2 Background Material

2.1 SSIM

The function SSIM measures the similarity between an 'ideal' discrete image x and a distorted version of it y based on three components, luminance, contrast and structure [2,6]. Let μ_x, μ_y be the mean intensity of images x and y each of size N as

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i \quad , \quad \mu_y = \frac{1}{N}\sum_{i=1}^{N} y_i.$$

Let σ_x, σ_y and σ_{xy} be the standard deviation of x, standard deviation of y, and covariance between x and y respectively given by $\sigma_x = \left(\frac{1}{N-1}\sum_{i=1}^{N}(x_i-\mu_x)^2\right)^{\frac{1}{2}}$, $\sigma_y = \left(\frac{1}{N-1}\sum_{i=1}^{N}(y_i-\mu_y)^2\right)^{\frac{1}{2}}$ and $\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i-\mu_x)(y_i-\mu_y)$. The luminance comparison is defined as

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2+\mu_y^2+C_1},$$

the contrast comparison is calculated as

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2+\sigma_y^2+C_2},$$

and the structure comparison is defined as

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3},$$

where the constants $C1 = (0.01)^2$ and $C2 = (0.03)^2$ are included to avoid instability when factors $(\mu_x^2+\mu_y^2)$, $(\sigma_x^2+\sigma_y^2)$ or $\sigma_x\sigma_y$ are close to zero. Combining the three comparison components, we obtain the SSIM index between images x and y in a scale of 0 to 1 as

$$SSIM(x,y) = [l(x,y)]^\alpha \cdot [c(x,y)]^\beta \cdot [s(x,y)]^\gamma$$

where $\alpha, \beta, \gamma > 0$ are parameters used to assign a weight to each of the three comparison terms. Setting $\alpha = \beta = \gamma = 1$ and $C_3 = \frac{C2}{2}$, we obtain the expression of the SSIM that will be used in this manuscript

$$SSIM(x,y) = \left(\frac{2\mu_x\mu_y + C_1}{\mu_x^2+\mu_y^2+C_1}\right)\left(\frac{2\sigma_{xy} + C_2}{\sigma_x^2+\sigma_y^2+C_2}\right).$$

2.2 MMSIM

The SSIM is applied locally rather than globally, obtaining a local SSIM for each point of an image using a sliding Gaussian window of fixed size. This provides a map which provides information about how the image quality varies spatially. A global SSIM can be obtained by calculating its average. In this case, mean SSIM (MSSIM) is introduced [1,6] as

$$MSSIM(x,y) = \frac{1}{N}\sum_{j=1}^{N} SSIM(x_j, y_j), \tag{1}$$

where N is the number of local windows in the image and x_j and y_j are the j-th sample sliding square windows, also referred to as patches, in images x and y.

2.3 MSSIM as Data Fidelity in Imaging Inverse Problems

As discussed in [5], we typically infer a measurement image $y = H(x) + n$ that is related to an ideal image x, through a degradation operator H and some additive noise term n. In general we assume that the operator H is known or can be approximated.

To obtain an approximation of the ideal image x, given y we can solve a minimization problem

$$\underset{x}{\operatorname{argmin}}\, [1 - MSSIM(H(x), y)] + \alpha[R(x)], \tag{2}$$

where H is the degradation operator, $R(x)$ is a regularization term, and α is its corresponding regularization parameter. Notice that the new data fidelity term $[1 - MSSIM(H(x), y)]$ replaces the typical Sum of Squared Differences of $H(x)$ and y in inverse problems as motivated by [5]. In this manuscript, for simplicity we consider the particular case where $H = I$, that is, denoising a given measurement image y and reconstructing the ideal image x, where $y = x + n$. In this case, our minimization problem becomes

$$\underset{x}{\operatorname{argmin}}\, [1 - MSSIM(x, y)] + \alpha[R(x)]. \tag{3}$$

The same technique that we will develop in this manuscript for computing derivative of MSSIM can be applied to any inverse problem addressed via the minimization in Eq. (2).

3 Computing the MSSIM Derivative

The SSIM derivative was computed in the original SSIM papers [6,7] although full details were not given. It was also computed as an essential step in the Maximum Differentiation (MAD) competition [8]. It was later extended in solving optimization problems, e.g. see [3].

In order to apply gradient-based techniques to solve the minimization problem in Eq. (3), we need to find the MSSIM derivative. In this section, we express this derivative efficiently using convolutions. Recall that convolution operator $*$ of two functions f and w is defined as

$$g(x) = \big(f * w\big)(x) = \sum_n f(n)w(x-n).$$

Theorem 1. *The partial derivative of MSSIM between two images x and y with respect to $x(q)$, i.e, the intensity of image x as location q can be expressed using convolutions as*

$$\frac{\partial MSSIM(x,y)}{\partial x(q)} = \frac{2}{N}\Big((h * G_\sigma)(q) + y(q)(f * G_\sigma)(q) - (g * G_\sigma)(q) - x(q)(j * G_\sigma)(q) + (k * G_\sigma)(q)\Big) \quad (4)$$

where N is the total number of pixels (also corresponding to the number of patches) in the image, and for every pixel location p

$$\begin{aligned} h(p) &= cs(p)\Big(\frac{\mu_y - \mu_x l(p)}{\mu_x^2 + \mu_y^2 + C_1}\Big), \\ f(p) &= \frac{l(p)}{\sigma_x^2 + \sigma_y^2 + C_2}, \\ g(p) &= \frac{l(p)\mu_y(p)}{\sigma_x^2 + \sigma_y^2 + C_2}, \\ j(p) &= \frac{l(p)cs(p)}{\sigma_x^2 + \sigma_y^2 + C_2}, \\ k(p) &= \frac{l(p)cs(p)\mu_x(p)}{\sigma_x^2 + \sigma_y^2 + C_2}. \end{aligned}$$

Proof. First, we find the SSIM derivatives as given in [9]. We can express the SSIM between images x and y at a local patch centered at pixel p as

$$SSIM\,(x(p), y(p)) = \left(\frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}\right) \cdot \left(\frac{2\sigma_{xy} + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}\right) = l(p)cs(p). \quad (5)$$

where means and standard deviations are computed using G_σ which is a Gaussian filter of standard deviation σ,

$$\mu_x(p) = G_\sigma * P_x, \quad (6)$$
$$\sigma_x^2(p) = G_\sigma * P_x^2 - \mu_x^2(p), \quad (7)$$
$$\sigma_{xy}(p) = G_\sigma * (P_x \cdot P_y) - \mu_x(p)\mu_y(p), \quad (8)$$

where P_x and P_y are patches in x and y centered at pixel p respectively, '$*$' denotes the convolution operator and '$\cdot$' is point-wise multiplication. The values of $\mu_y(p)$ and $\sigma_y^2(p)$ can be computed similar to Eqs. (6) and (7) replacing x with y. We need to compute the derivative at pixel p with respect to every pixel location q in every patch. As shown in [9],

$$\frac{\partial SSIM(x(p), y(p))}{\partial x(q)} = \frac{\partial l(p)}{\partial x(q)} cs(p) + l(p) \frac{\partial cs(p)}{\partial x(q)},$$

where

$$\frac{\partial l(p)}{\partial x(q)} = \frac{\partial l(p)}{\partial \mu_x} \cdot \frac{\partial \mu_x}{\partial x(q)} = 2G_\sigma(p-q) \cdot \left(\frac{\mu_y - \mu_x \cdot l(p)}{\mu_x^2 + \mu_y^2 + C_1} \right), \tag{9}$$

and

$$\frac{\partial cs(p)}{\partial x(q)} = \frac{2G_\sigma(p-q)}{\sigma_x^2 + \sigma_y^2 + C_2} \Big(\big(y(q) - \mu_y\big) - cs(p)\big(x(q) - \mu_x\big) \Big). \tag{10}$$

We apply these derivative approximations to compute the derivative of MSSIM. The partial derivatives of MSSIM are computed as

$$\frac{\partial MSSIM(x, y)}{\partial x(q)} = \frac{1}{N} \sum_p \frac{\partial l(p)}{\partial x(q)} cs(p) + l(p) \frac{\partial cs(p)}{\partial x(q)}, \tag{11}$$

where N is the total number of pixels, also patches, in the image.

Now we expand both parts of the derivatives and show that it can be expressed as a product of convolutions. The first expression on the right hand side of Eq. (11) can be written as

$$\begin{aligned}
&\frac{1}{N} \sum_p \frac{\partial l(p)}{\partial x(q)} cs(p) \\
&= \frac{1}{N} \sum_p 2G_\sigma(p-q) \left(\frac{\mu_y - \mu_x l(p)}{\mu_x^2 + \mu_y^2 + C_1} \right) cs(p) \\
&= \frac{2}{N} \sum_p cs(p) \left(\frac{\mu_y - \mu_x l(p)}{\mu_x^2 + \mu_y^2 + C_1} \right) G_\sigma(p-q) \\
&= \frac{2}{N} \sum_p h(p) G_\sigma(p-q) \\
&= \frac{2}{N} \big(h * G_\sigma\big)(q),
\end{aligned}$$

where $h(p) = cs(p)\big(\frac{\mu_y - \mu_x l(p)}{\mu_x^2 + \mu_y^2 + C_1}\big)$. The second expression can be written as

$$\begin{aligned}
&\frac{1}{N} \sum_p l(p) \frac{\partial cs(p)}{\partial x(q)} \\
&= \frac{1}{N} \sum_p \frac{2l(p) G_\sigma(p-q)}{\sigma_x^2 + \sigma_y^2 + C_2} \Big(\big(y(q) - \mu_y\big) - cs(p)\big(x(q) - \mu_x\big) \Big) \\
&= \frac{2}{N} \big(A + B + C + D\big),
\end{aligned}$$

where

$$\begin{aligned} A &= \sum_p \frac{l(p)}{\sigma_x^2+\sigma_y^2+C_2} G_\sigma(p-q)y(q) \\ &= y(q)\sum_p \frac{l(p)}{\sigma_x^2+\sigma_y^2+C_2} G_\sigma(p-q) \\ &= y(q)\big(f * G_\sigma\big)(q), \end{aligned}$$

with $f(p) = \frac{l(p)}{\sigma_x^2+\sigma_y^2+C_2}$,

$$\begin{aligned} B &= -\sum_p \frac{l(p)}{\sigma_x^2+\sigma_y^2+C_2} G_\sigma(p-q)\mu_y(p) \\ &= -\sum_p \frac{l(p)\mu_y(p)}{\sigma_x^2+\sigma_y^2+C_2} G_\sigma(p-q) \\ &= -\big(g * G_\sigma\big)(q), \end{aligned}$$

with $g(p) = \frac{l(p)\mu_y(p)}{\sigma_x^2+\sigma_y^2+C_2}$,

$$\begin{aligned} C &= -\sum_p \frac{l(p)}{\sigma_x^2+\sigma_y^2+C_2} G_\sigma(p-q)cs(p)x(q) \\ &= -x(q)\sum_p \frac{l(p)cs(p)}{\sigma_x^2+\sigma_y^2+C_2} G_\sigma(p-q) \\ &= -x(q)\big(j * G_\sigma\big)(q), \end{aligned}$$

with $j(p) = \frac{l(p)cs(p)}{\sigma_x^2+\sigma_y^2+C_2}$, and

$$\begin{aligned} D &= \sum_p \frac{l(p)}{\sigma_x^2+\sigma_y^2+C_2} G_\sigma(p-q)cs(p)\mu_x(p) \\ &= \sum_p \frac{l(p)cs(p)\mu_x(p)}{\sigma_x^2+\sigma_y^2+C_2} G_\sigma(p-q) \\ &= \big(k * G_\sigma\big)(q), \end{aligned}$$

with $k(p) = \frac{l(p)cs(p)\mu_x(p)}{\sigma_x^2+\sigma_y^2+C_2}$. $\square$

The above Theorem can essentially be used to compute the derivative of the MSSIM using convolutions that will be referred to as our proposed algorithm.

4 Efficiency

In order to show the efficiency of our proposed algorithm, we analyze its computational complexity. We compare two different paths to compute the MSSIM derivative.

First path evaluating the derivative directly by computing each expression of the sum in Eq. (11) and adding all of these terms, and second path using convolutions as shown in Eq. (4). Suppose the size of the image is N and the size of each patch is m. Hence we have N patches of size m.

In the first path, we analyze the expression in Eq. (11). For every patch, the calculation consists of two multiplications and an addition of terms depending on μ_x, μ_y, σ_x, σ_y and $\sigma_{x,y}$, which are calculated with convolutions as shown in Eqs. (6, 7 and 8). We know that the computational cost of the convolution using Fast Fourier Transform (FFT) is $\mathcal{O}(N \log N)$, and both terms (Eqs. 9 and 10) are calculated with respect to every pixel q in the patch centered at p. Therefore the cost of the calculation is $\mathcal{O}(m\ N \log N)$.

Now using the second path, we analyze the expression in Eq. (4). The calculation consists of two additions, 2 subtractions and 3 multiplications of convolutions and terms depending again on μ_x, μ_y, σ_x, σ_y and $\sigma_{x,y}$. Hence the cost of calculating the derivative using convolutions is $\mathcal{O}(N \log N)$.

We can conclude that the larger the patch-size m, the larger is the difference between the two computational complexities. As $m \to N$, the cost of the calculation of the MSSIM derivative gets closer to $\mathcal{O}(N^2 \log N)$, and in general using convolutions via our proposed Theorem provides a more efficient way to calculate the derivatives.

5 Experiments and Results

In this Section, we consider the total variation (TV) denoising in which we use the MSSIM as the data fidelity term. Variational calculus tells us that the minimum of the (convex) function

$$E = \int_{\Omega} \mathbf{g}(x, y, u, u_x, u_y)\, d\boldsymbol{x}$$

can be found by setting the variational derivative equal to zero

$$\nabla E = \frac{d\mathbf{g}}{du} - \frac{d}{dx}\frac{d\mathbf{g}}{du_x} - \frac{d}{dy}\frac{d\mathbf{g}}{du_y} = 0.$$

To solve this equation numerically, we evolve u to the state minimizing E

$$\frac{du}{dt} = -\nabla E.$$

Discretizing this equation yields the steepest descent method

$$u^{n+1} = u^n + \Delta t \left[-\nabla E(u^n)\right]. \tag{12}$$

The total variation of an image $u(x, y)$ is

$$\int_{\Omega} \|\nabla u\|\, d\boldsymbol{x}.$$

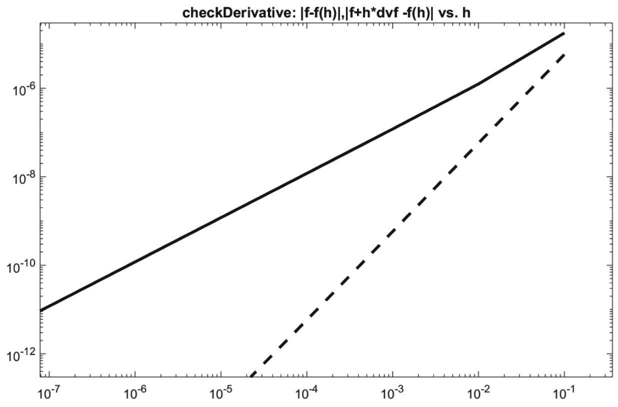

Fig. 1. The result of checkDerivative for the implementation of MSSIM derivative: linear decay (solid line) for $|\mathbf{f} - \mathbf{f}(\mathtt{h})|$ and quadratic decay (dashed line) for $|\mathbf{f} + \mathtt{h} * dv\mathbf{f} - \mathbf{f}(\mathtt{h})|$ up to machine precision is shown on a logarithmic scale.

To minimize total variation, we could pick $\mathbf{g} = \|\nabla u\|$, but evolving such a system would eventually lead to a flat image $u = const$. An additional term is needed to determine the balance minimizing the variation with keeping the evolved u similar to the noisy measurement w. Traditionally we can add a data fidelity term $\int_\Omega \lambda(u-w)^2\, d\boldsymbol{x}$, so that E is given by

$$E = \int\limits_\Omega \|\nabla u\|\, d\boldsymbol{x} + \lambda \int\limits_\Omega (u-w)^2\, d\boldsymbol{x}. \tag{13}$$

Alternatively, we use the new data fidelity term $\int_\Omega MSSIM(u,w)\, d\boldsymbol{x}$, and let $\mathbf{g} = \|\nabla u\| + \lambda MSSIM(u,w)$ as motivated in [5]

$$E = \int\limits_\Omega \|\nabla u\|\, d\boldsymbol{x} + \lambda \int\limits_\Omega MSSIM(u,w)\, d\boldsymbol{x}. \tag{14}$$

To minimize Eq. (14), we first obtain ∇E

$$\begin{aligned}\nabla E &= \frac{d\mathbf{g}}{du} - \frac{d}{dx}\frac{d\mathbf{g}}{du_x} - \frac{d}{dy}\frac{d\mathbf{g}}{du_y} \\ &= \lambda \frac{\partial MSSIM(u,w)}{\partial u} - \frac{u_{xx}u_y^2 - 2u_x u_y u_{xy} - u_{yy}u_x^2}{(u_x^2+u_y^2)^{3/2}},\end{aligned} \tag{15}$$

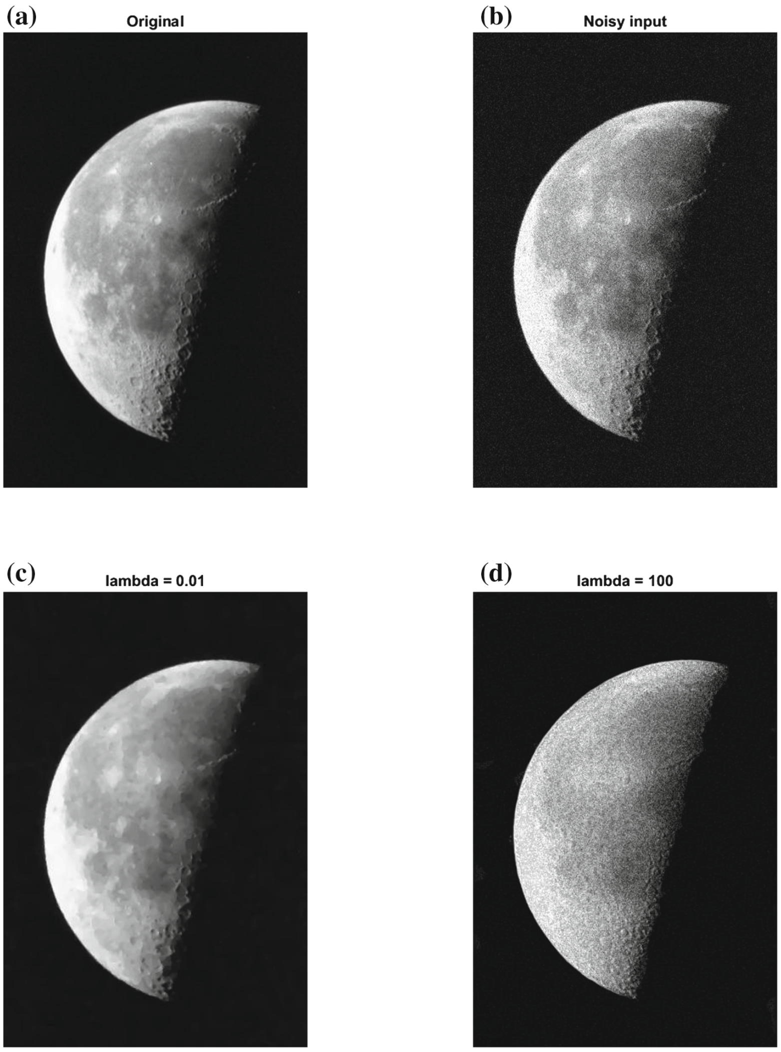

Fig. 2. (a) The original moon image. (b) Noisy input, SSIM: 0.29, PSNR: 24.77 (c) TV denoising with $\lambda = 0.01$, SSIM: 0.83, PSNR: 34:84. (d) TV denoising with $\lambda = 100$, SSIM: 0.39, PSNR: 23.92

where we used $u_{xy} = u_{yx}$. Then we evolve the PDE with a small nonzero ϵ added to the denominator of the fraction term so that the term does not become singular

$$\begin{aligned}\frac{du}{dt} &= -\nabla E \\ &= \frac{u_{xx}u_y^2 - 2u_x u_y u_{xy} - u_{yy}u_x^2}{\epsilon + (u_x^2 + u_y^2)^{3/2}} - \lambda \frac{\partial MSSIM(u, w)}{\partial u}.\end{aligned}$$

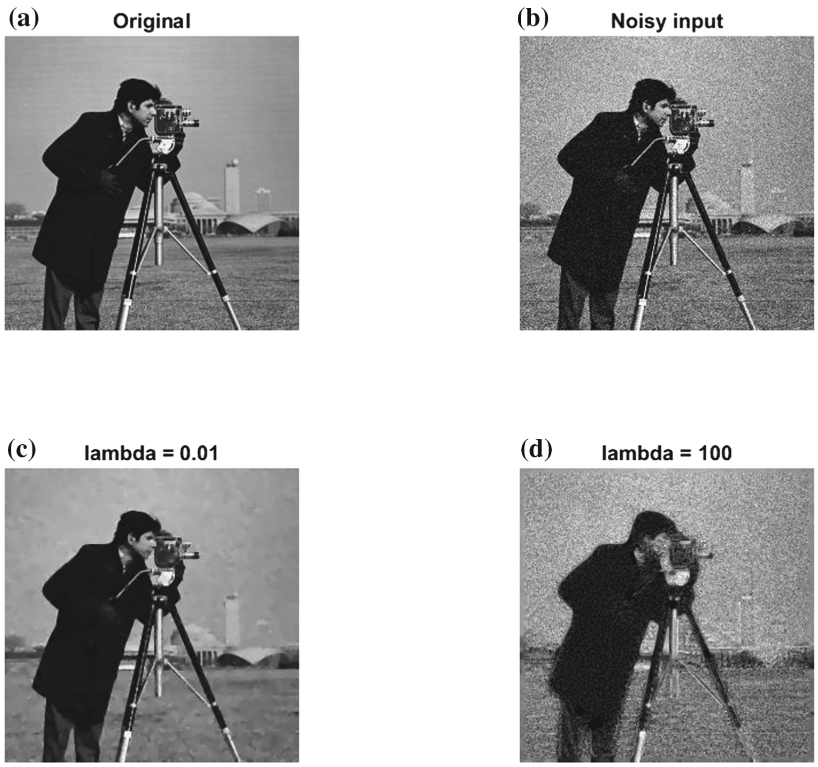

Fig. 3. (a) The original cameraman image. (b) Noisy input, SSIM: 0.44, PSNR: 23.32. (c) TV denoising with $\lambda = 0.01$, SSIM: 0.82, PSNR: 28.02. (d) TV denoising with $\lambda = 100$, SSIM: 0.08, PSNR: 18.45.

The discretized version is (from Eqs. (12) and (15))

$$\begin{aligned} u^{(n+1)} &= u^{(n)} + \Delta t\,[-\nabla E] \\ &= u^{(n)} + \Delta t \left[\frac{u_{xx}^{(n)} u_y^{(n)2} - 2u_x^{(n)} u_y^{(n)} u_{xy}^{(n)} - u_{yy}^{(n)} u_x^{(n)2}}{\epsilon + (u_x^{(n)2} + u_y^{(n)2})^{3/2}} - \lambda \frac{\partial MSSIM(u^{(n)}, w)}{\partial u^{(n)}} \right]. \end{aligned} \tag{16}$$

One of the many traps in optimization is working with an erroneous derivative. The following test provides a simple way of checking the implementation of a derivative. To this end, let $\mathbf{f}$ be a multivariate function $\mathbf{f} : \mathbb{R}^n \longmapsto \mathbb{R}$ and let $\mathtt{v} \in \mathbb{R}^n$ be an arbitrary vector in the Taylor expansion $\mathbf{f}(\mathtt{x} + \mathtt{h}\mathtt{v}) = \mathbf{f}(\mathtt{x}) + \mathtt{h}d\mathbf{f}(\mathtt{x})\mathtt{v} + \mathcal{O}(\mathtt{h}^2)$. A matrix $\mathbf{A}$ is the derivative of $\mathbf{f}$ if and only if the difference $\|\mathbf{f}(\mathtt{x} + \mathtt{h}\mathtt{v}) - \mathbf{f}(\mathtt{x}) - \mathtt{h}\mathbf{A}\mathtt{v}\|$ is essentially quadratic in $\mathtt{h}$. The function checkDerivative (See, e.g., [4]) computes this difference. Here we apply the checkDerivative function to our computed MSSIM derivative and show the result in Fig. 1. The result is consistent as the difference is quadratic with respect to $\mathtt{h}$.

To observe how the TV denoising algorithm with MSSIM data fidelity works (see Eq. (16)), we added zero-mean additive white Gaussian noise to the images 'Moon' (Fig. 2) and 'Cameraman' (Fig. 3) for various parameters λ and calculated the SSIM and PSNR values of the results.

As expected, we can perform denoising using the new MSSIM data fidelity term for a proper value of the regularization parameter λ, e.g., $\lambda = 0.01$ for these images. For a complete list of parameters used along with the checkDerivative results, please refer to our source code which is publicly available on GitHub at https://github.com/ImagingLab/MSSIM.

6 Conclusion

In this paper, we introduced a computationally efficient algorithm to calculate the MSSIM derivative used in imaging inverse problems, e.g, image denoising.

The derivative calculation was rigorously obtained based on convolutions in Theorem 1. We also presented preliminary computational experiments to validate the calculation of the MSSIM derivative and applied it to Total Variation denoising method as a data fidelity term. As mentioned, we focused on computing the derivative aimed at denoising (2) problem. To extend the technique for solving (3), chain rule can be used to compute the derivative in a similar way. This will generalize to address a variety of inverse problems including deblurring and super-resolution with a new data fidelity term.

Acknowledgments. I.M. is a visiting student from the Universidad de Málaga, Spain. This research was supported in part by an NSERC Discovery Grant for M.E.

References

1. Brunet, D., Vrscay, E.R., Wang, Z.: On the mathematical properties of the structural similarity index. IEEE Trans. Image Process. **21**(4), 1488–1499 (2012)
2. Hore, A., Ziou, D.: Image quality metrics: PSNR vs. SSIM. In: 2010 20th International Conference on Pattern Recognition (ICPR), pp. 2366–2369. IEEE (2010)
3. Ma, K., Duanmu, Z., Yeganeh, H., Wang, Z.: Multi-exposure image fusion by optimizing a structural similarity index. IEEE Trans. Comput. Imaging **4**(1), 60–72 (2017)
4. Modersitzki, J.: FAIR: Flexible Algorithms for Image Registration, vol. 6. SIAM, New Delhi (2009)
5. Otero, D., Vrscay, E.R.: Solving optimization problems that employ structural similarity as the fidelity measure. In: Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV), page 1. The Steering Committee of the World Congress in Computer Science, Computer (2014)
6. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
7. Wang, Z., Simoncelli, E.P.: Stimulus synthesis for efficient evaluation and refinement of perceptual image quality metrics. In: Human Vision and Electronic Imaging IX, vol. 5292, pp. 99–109. International Society for Optics and Photonics (2004)

8. Wang, Z., Simoncelli, E.P.: Maximum differentiation (MAD) competition: a methodology for comparing computational models of perceptual quantities. J. Vis. **8**(12), 8–8 (2008)
9. Zhao, H., Gallo, O., Frosio, I., Kautz, J.: Loss functions for image restoration with neural networks. IEEE Trans. Comput. Imaging **3**(1), 47–57 (2017)