IMPROVING DENOISING FILTERS BY OPTIMAL DIFFUSION[*]

Hossein Talebi and Peyman Milanfar

Department of Electrical Engineering
University of California, Santa Cruz, CA, 95064

ABSTRACT

Kernel based methods have recently been used widely in image denoising. Tuning the parameters of these algorithms directly affects their performance. In this paper, an iterative method is proposed which optimizes the performance of *any* kernel based denoising algorithm in the mean-squared error (MSE) sense, even with arbitrary parameters. In this work we estimate the MSE in each image patch, and use this estimate to guide the iterative application to a stop, hence leading to improve performance. We propose a new estimator for the risk (i.e. MSE) which is different than the often-employed SURE method. We illustrate that the proposed risk estimate can outperform SURE in many instances.

***Index Terms*—** Image Denoising, Anisotropic Diffusion, Data-dependent Filtering, Risk Estimation

1. INTRODUCTION

In the past few years, non-parametric restoration methods have become extremely popular. These new algorithms are mostly patch-based, and also employ local and non-local similarities in the signals [1, 2, 3]. Let us consider the measurement model for the denoising problem:

$$y_i = z_i + e_i, \qquad \text{for } i = 1, ..., n, \tag{1}$$

where $z_i = z(\mathbf{x}_i)$ is the underlying image at position $\mathbf{x}_i = [x_{i,1}, x_{i,2}]^T$, y_i is the noisy pixel value, and e_i is zero-mean, white noise with variance σ^2. The problem of denoising is to recover the set of underlying samples $\mathbf{z} = [z(\mathbf{x}_1), ..., z(\mathbf{x}_n)]^T$. The complete measurement model for the denoising problem in vector notation is:

$$\mathbf{y} = \mathbf{z} + \mathbf{e} \tag{2}$$

The estimate of the underlying signal $\mathbf{z}$ is found using a weighted least square framework [4] in which the weight function $K(.)$ measures the similarity between samples y_i and y_j at respective $\mathbf{x}_i$ and $\mathbf{x}_j$ positions. Perhaps the most well known kernel in this class is the Bilateral (BL) filter [2], which smooths images by means of a nonlinear combination of nearby image values. The method combines pixel values based on both their geometric closeness and their photometric similarity. This kernel can be expressed in a separable fashion as follows:

$$K_{ij} = \exp\left\{\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{h_x^2} + \frac{-(y_i - y_j)^2}{h_y^2}\right\}, \tag{3}$$

in which h_x and h_y are control parameters. The BL kernel also can be thought of as weighted Euclidean distance between the vectors $(\mathbf{x}_i, y_i)$ and $(\mathbf{x}_j, y_j)$. The Non-Local Means (NLM) [3] is another very popular data-dependent filter which closely resembles the bilateral filter except that the photometric similarity is captured in a patch-wise manner:

$$K_{ij} = \exp\left\{\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{h_x^2} + \frac{-\|\mathbf{y}_i - \mathbf{y}_j\|^2}{h_y^2}\right\}, \tag{4}$$

where $\mathbf{y}_i$ and $\mathbf{y}_i$ are patches centered at y_i and y_j respectively. In theory (though not in actual practice,) the NLM kernel has just the patch-wise photometric distance ($h_x \longrightarrow \infty$). More recently, the LARK kernel [1] was introduced which exploits the geodesic distance based on estimated gradients:

$$K_{ij} = \exp\{-(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{C}_{ij} (\mathbf{x}_i - \mathbf{x}_j)\}, \tag{5}$$

in which $\mathbf{C}_{ij}$ is a local covariance matrix of the pixel gradients computed from the given data. The gradient is computed from the noisy measurements y_j in a patch around $\mathbf{x}_i$. Robustness to noise and perturbations of the data is an important advantage of the LARK similarity metric. In general, all of these restoration algorithms are based on the same framework in which some data-adaptive weights are assigned to each pixel contributing to the filtering. The matrix-vector multiplication form of the denoising filter where $\widehat{\mathbf{z}}$ and $\mathbf{W}$ denote the filtered signal and the matrix of filter weights, respectively, is as follows:

$$\widehat{\mathbf{z}} = \begin{bmatrix} \mathbf{w}_1^T \\ \mathbf{w}_2^T \\ \vdots \\ \mathbf{w}_n^T \end{bmatrix} \mathbf{y} = \mathbf{W}\mathbf{y}, \tag{6}$$

This filter is given by the weights defined by any one of the kernels discussed above, where each row of the filter $\mathbf{W}$ can be expressed as:

$$\mathbf{w}_j = \frac{1}{\sum_{i=1}^{n} K_{ij}} [K_{1j}, K_{2j}, \ldots, K_{nj}]^T. \tag{7}$$

[*]This work was supported by AFOSR Grant FA9550-07-1-0365 and NSF Grant CCF-1016018.

W is a positive row-stochastic matrix (every row sums up to one). This matrix is not generally symmetric, though it has real, positive eigenvalues [4]. The Perron-Frobenius theory describes the spectral characteristics of this matrix. In particular, the eigenvalues of **W** satisfy $0 \leq \lambda_i \leq 1$; the largest one is uniquely equal to one, ($\lambda_1 = 1$) while the corresponding eigenvector is $\mathbf{v}_1 = \frac{1}{\sqrt{n}}[1, 1, ..., 1]^T$. The last property implies that a flat image stays unchanged after filtering by **W**. Although **W** is not a symmetric matrix in general, it can be closely approximated with a symmetric positive definite matrix [4]. The symmetrized **W** should also stay row-stochastic, which means we get a symmetric positive definite matrix which is doubly (i.e., row- *and* column-) stochastic. The symmetric **W** enables us to do the following analysis. The spectrum of **W** specifies the effect of the filter on the noisy signal. Its eigen-decomposition can be written as:

$$\mathbf{W} = \mathbf{V}\mathbf{S}\mathbf{V}^T, \tag{8}$$

where $\mathbf{S} = \text{diag}[\lambda_1, ..., \lambda_n]$ contains the eigenvalues in decreasing order $0 \leq \lambda_n \leq ... < \lambda_1 = 1$, and **V** is an orthogonal matrix $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_n]$ containing the respective eigenvectors of **W** in its columns.

The matrix **W** is in general a function of the given data vector **y**; however, computing **W** from **y** usually involves a pre-filtering stage in which a pilot estimate of the latent image is computed. This pre-denoising guides the filter to depend more closely on the signal **z** and less on the noise [4].

2. ITERATIVE DATA-DEPENDENT FILTERING

It is possible to improve the performance of a given filter by applying it multiple times. More explicitly, going back to the definition of the kernels, although the kernels are adaptive to different parts of image, still there is a need to take into account the signal-to-noise (SNR) ratio of those parts. In other words, depending on the SNR, *over*- or *under*- denoising might happen in different parts of the image. The iterative framework is given by:

$$\widehat{\mathbf{z}}_k = \mathbf{W}^k\mathbf{y}, \tag{9}$$

where each application of **W** can be interpreted as one step of anisotropic diffusion [4, 5] with the filter **W**. Diffusion filtering gradually removes noise in each iteration, but also takes away latent details from the underlying signal. Choosing a small iteration number k preserves the underlying structure, but also does little denoising. On the other hand, a large k tends to over-smooth and remove noise and high frequency details at the same time. The optimal stopping time $\widehat{k}$ can be expressed as:

$$\widehat{k} = \arg\min_k \ \text{MSE}_k, \tag{10}$$

where MSE_k is the mean-squared error of the filtering model (9). In the following, we discuss estimation of the MSE_k in the context of data-dependent filtering. First, we aim to find the best iteration number for each patch and then, we extend this denoising approach to the whole image.

3. ESTIMATION OF RISK

To estimate the MSE_k for each patch, we propose a plug-in risk estimator. This method is biased and works based on a"pilot" estimate of the latent image. We also derive Stein's unbiased risk estimator (SURE) [6] for the data dependent filtering scheme and compare it with the proposed one. While [7] also uses SURE estimator to optimize the NLM kernel parameters, we illustrate that (1) the plug-in estimator can be superior for the same task, and (2) the adaptation approach can be extend to be spatially varying.

3.1. Plug-in Risk Estimator

The filter in the iterative model (9) can be decomposed as:

$$\mathbf{W}^k = \mathbf{V}\mathbf{S}^k\mathbf{V}^T, \tag{11}$$

in which $\mathbf{S}^k = \text{diag}[\lambda_1^k, ..., \lambda_n^k]$. It is important to note that despite the earlier interpretation of k as a discrete step, the spectral decomposition of $\mathbf{W}^k$ makes it clear that in practice, k can be any positive real number. In actual implementation, the filter can be applied with modified eigenvalues regardless of the value of k. Considering the eigen-decomposition, the image **z** can be written in the column space of **V** as $\mathbf{b} = \mathbf{V}^T\mathbf{z}$, where $\mathbf{b} = [b_1, b_2, ..., b_n]^T$, and $\{b_i^2\}$ denote the signal energy distribution over all the modes. As shown in [4] the iterative estimator $\widehat{\mathbf{z}}_k = \mathbf{W}^k\mathbf{y}$ has the following MSE:

$$\text{MSE}_k = \sum_{i=1}^{n}(\lambda_i^k - 1)^2 b_i^2 + \sigma^2\lambda_i^{2k}. \tag{12}$$

This closed form of MSE is expressed succinctly in terms of the filter, the latent signal and the additive noise variance. In [8] we proposed an estimation of $\{b_i\}$ in which the latent image **z** was estimated by an optimal"pre-processing" stage. Here we assume that the noisy image is "pre-filtered" ($\widetilde{\mathbf{z}} = \mathbf{W}\mathbf{y}$) and then the components of the filter and the latent image are estimated. As such, the proposed plug-in MSE estimate can be expressed as:

$$\widehat{\text{MSE}}_k = \sum_{i=1}^{n}(\lambda_i^k - 1)^2 \widetilde{b}_i^2 + \sigma^2\lambda_i^{2k}, \tag{13}$$

where $\widetilde{\mathbf{b}} = \mathbf{V}^T\widetilde{\mathbf{z}}$ is the estimated signal energy distribution from the pre-filtered image $\widetilde{\mathbf{z}}$. Next, for the sake of comparison, the SURE estimator is discussed.

3.2. SURE Estimator

Another way to estimate MSE_k is to use the SURE estimator [6]. Denoting $F(\mathbf{y})$ as an estimate of the latent signal **z**, for estimating $\text{E}(\|\mathbf{y} - F(\mathbf{y})\|^2)$ from **y**, the SURE risk estimator is:

$$\text{SURE}(\mathbf{y}) = \|\mathbf{y} - F(\mathbf{y})\|^2 + 2\sigma^2\text{div}(F(\mathbf{y})) - n\sigma^2, \tag{14}$$

where $\text{div}(F(\mathbf{y})) \equiv \sum_i \frac{\partial F_i(\mathbf{y})}{\partial y_i}$. Under the additive Gaussian noise assumption, this random variable is an unbiased estimate of the MSE. More specifically, the SURE for a filter of the type (9) is given by:

$$\mathrm{SURE}(\mathbf{y}) = \|(\mathbf{I} - \mathbf{W}^k)\mathbf{y}\|^2 + 2\sigma^2 \mathrm{tr}(\mathbf{W}^k) - n\sigma^2. \quad (15)$$

Considering the eigen-decomposition of the filter, replacing $\mathbf{y}$ with $\mathbf{V}\breve{\mathbf{b}}$ (where $\breve{\mathbf{b}}$ is the energy distribution of the *noisy* signal over the eigenvectors) after some simplifications, we have

$$\mathrm{SURE}(\mathbf{y}) = \sum_{i=1}^{n} (\lambda_i^k - 1)^2 \breve{b}_i^2 + 2\sigma^2 \lambda_i^k - \sigma^2. \quad (16)$$

It is easy to show that the expected value of SURE($\mathbf{y}$) will replace $\breve{b}_i^2$ with $b_i^2 + \sigma^2$, which after simplification indeed yields (12).

We have discussed two MSE estimators in (13) and (16) for finding the optimal stopping time $\widehat{k}$ and then the best estimate of each patch as $\widehat{\mathbf{z}} = \mathbf{W}^{\widehat{k}}\mathbf{y}$. The next step is to extend the denoising procedure to the whole image. To avoid the blockiness effect at the patch bounderies, we overlap the patches, so there should be an aggregation strategy. In the following, after discussing the aggregation step, we aim to compare performances of the two MSE estimators for the existing kernels.

4. AGGREGATION STRATEGY

So far, we have found the best estimate for the iteration number in each patch. Our optimized per-patch filtering can be expressed as:

$$\widehat{\mathbf{z}}_j = \mathbf{W}^{\widehat{k}_j} \mathbf{y}_j, \quad (17)$$

in which $\mathbf{y}_j$ and $\widehat{\mathbf{z}}_j$ are the j-th noisy and denoised patch, respectively, and $\widehat{k}_j$ denotes the ideal stopping time for this patch. As a result of the overlapped patches, multiple estimates are obtained for each pixel. What we have now is a vector of measurements of the pixel z_l which need to be weighted and form the final denoised pixel $\widehat{z}_l$. Fig.1 illustrates an example of three overlapping patches and the computed multiple estimates in each of them. The aggregation method in [9] combines the multiple estimates in an LMMSE (liner minimum mean-squared-error) scheme that takes into account the relative confidence in each estimate as measured by the inverse of the estimator error variance. The error covariance of our proposed estimator is approximated as:

$$\mathbf{C}_e = \mathrm{cov}(\widehat{\mathbf{z}} - \mathbf{z}) = \mathrm{cov}(\mathbf{W}^{\widehat{k}_j}\mathbf{e}) = \sigma^2 \mathbf{W}^{2\widehat{k}_j}, \quad (18)$$

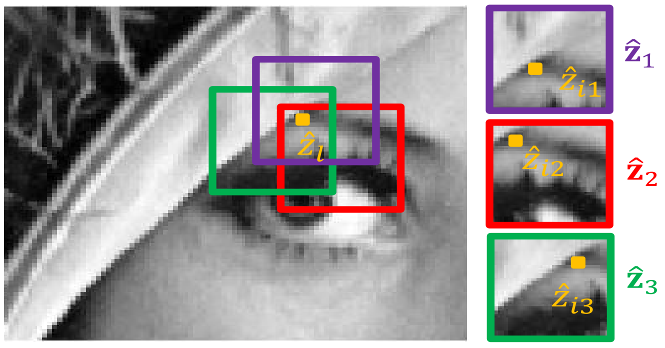

Fig. 1: Overlapping patches give multiple estimates for each pixel. Example of three overlapping patches $\widehat{\mathbf{z}}_1$, $\widehat{\mathbf{z}}_2$ and $\widehat{\mathbf{z}}_3$ give three estimates $\widehat{z}_{i1}$, $\widehat{z}_{i2}$ and $\widehat{z}_{i3}$ for computing the final denoised pixel $\widehat{z}_l$.

We denote $\widehat{z}_{ij}$ as the denoised estimate for the i-th pixel in the j-th patch $\widehat{\mathbf{z}}_j$. Then, the variance of the error associated with the i-th pixel estimate in the j-th patch, v_{ij}, is given by the i-th diagonal element of $\mathbf{C}_e$. Inverse of the estimator error variances $\{v_{ij}\}$, are the weights we use for the aggregation:

$$\widehat{z}_l = \sum_{j=1}^{N} \frac{\frac{\widehat{z}_{ij}}{v_{ij}}}{\sum_{j=1}^{N} \frac{1}{v_{ij}}}, \quad (19)$$

where N is the number of available estimates for the l-th pixel. Now we can implement the optimized iteration number per patch which adaptively minimizes MSE of the filtering in each block of the image.

5. SIMULATION RESULTS

The performance of our denoising method is evaluated on four 512×512 images: Peppers, Lena, Barbara, and Boat. In our simulations the patch size is set as 21×21 and in a Monte-Carlo simulation, 50 independent noise realizations were used. We varied k from 0 to 5 with 0.05 as the step size. The kernels in [2], [3] and [1] are used in our simulations. The control parameters in the Bilateral kernel are $h_x = 2\sqrt{2}$ and $h_y = 20\sqrt{2}\sigma$, the h_y in the NLM filter is fixed as 0.32σ and the smoothing parameter in LARK [1] is specified as 0.25σ.

Table 1 shows MSE results of the standard kernel (fixed parameters in BL, NLM, or LARK), SURE and the plug-in estimators. It can be seen that for Bilateral and Non-local means kernels, the plug-in estimator shows consistent improvement over both the standard estimate using the kernel, and the kernel iterated using $\widehat{k}$ from SURE. For the LARK

Table 1: MSE values for the application of various filters with fixed parameters (1st column); spatially adaptive iterations optimized with SURE estimator of section 3.2 (2nd column), and spatially adaptive iterations optimized with the plug-in risk estimator of section 3.1 (3rd column). ($\sigma^2 = 100$)

	Bilateral Kernel			
Images	Standard	SURE	Plug-in	Improvement
Peppers	27.85	30.52	**24.80**	3.05
Lena	28.89	30.42	**24.27**	4.62
Barbara	47.62	46.66	**42.80**	4.88
Boat	40.69	39.57	**36.26**	4.43
	Non-local Means Kernel			
Images	Standard	SURE	Plug-in	Improvement
Peppers	26.11	27.18	**23.69**	2.42
Lena	24.52	24.99	**21.62**	2.90
Barbara	31.82	30.28	**27.95**	3.87
Boat	34.94	34.36	**32.46**	2.50
	LARK Kernel			
Images	Standard	SURE	Plug-in	Improvement
Peppers	25.21	**22.81**	23.02	2.40
Lena	21.74	**18.99**	19.43	2.75
Barbara	52.41	**32.16**	38.76	20.25
Boat	33.10	**30.24**	31.62	2.86

Table 2: Performance of the plug-in estimator for the NLM kernel with different parameters ($\sigma^2 = 100$).

Images	$h_y^2 = 5$		$h_y^2 = 15$	
	Standard	Plug-in	Standard	Plug-in
Peppers	33.86	23.99	27.08	23.84
Lena	32.56	21.94	25.81	21.79
Barbara	42.43	28.61	33.85	28.20
Boat	41.71	32.81	39.02	32.91

kernel, the SURE method outperforms the plug-in estimator. Apparently, this performance difference occurs most noticeably for highly textured images such as Barbara. Overall, we can conclude that the plug-in estimator better fits "aggressive" kernel bases (kernels with eigenvalues $\{\lambda_i\}$ close to 0) such as NLM and Bilateral.

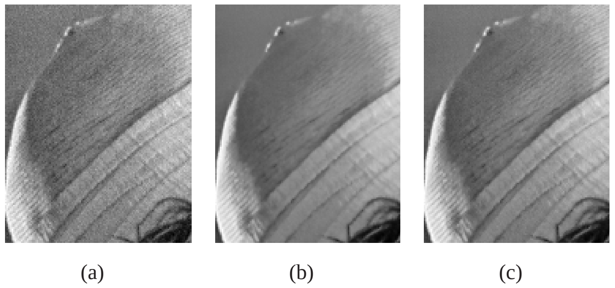

Fig. 2: NLM denoising of Lena corrupted by noise with $\sigma^2 = 25$: (a) Noisy, (b) Standard, MSE=11.91, (c) Plug-in, MSE=10.03

The effect of the parameter tuning is studied in Table 2. In this set of simulations, NLM kernel weights with different control parameter, h_y, were computed for each image and then fed to the plug-in estimator. As can be seen, across a large range of h_y, performance of the proposed estimator is quite stable and shows improvement over the standard kernel. From these results, we can see that it is possible to improve the performance of the standard kernel with an arbitrary parameter by employing the proper number of iteration with the proposed MSE estimator.

Fig.2 demonstrates the denoising results of Lena image obtained by the NLM kernel and the plug-in estimator. We can see that the proposed method provides better visual quality. Fig. 3 shows performance of the SURE estimator for the Barbara image. As it can be seen, both texture and smooth features of the image are preserved better than the LARK kernel.

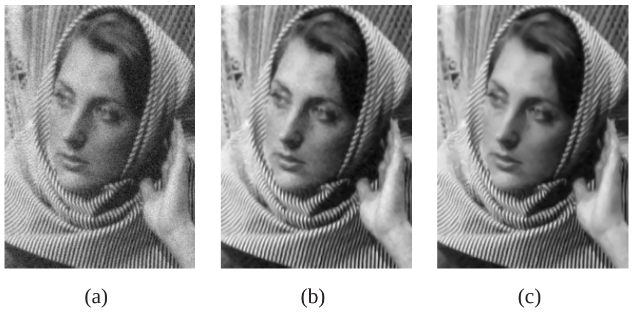

Fig. 3: LARK denoising of Barbara corrupted by noise with $\sigma^2 = 100$: (a) Noisy, (b) Standard, MSE=52.34, (c) SURE, MSE=32.12

6. CONCLUSION

We hav presented a framework for denoising by using data-dependent kernels. More specifically, by exploiting the best iteration number which minimizes MSE in each patch, the problems of performance of the previous kernel based methods in tuning parameters is mitigated and transformed to tuning a single parameter (k), which we achieve by estimating the risk patch-wise. The experimental results demonstrate that the proposed approach improves the kernel based methods performance in terms of both MSE and subjective visual quality.

7. REFERENCES

[1] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 349–366, February 2007.

[2] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," *Proc. of the 1998 IEEE International Conference of Computer Vision, Bombay, India*, pp. 836–846, January 1998.

[3] A. Buades, B. Coll, and J. M. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling and Simulation (SIAM interdisciplinary journal)*, vol. 4, no. 2, pp. 490–530, 2005.

[4] P. Milanfar, "A tour of modern image filtering," *To appear in IEEE Signal Processing Magazine 2012*, Talk available at http://tinyurl.com/moderntour.

[5] P. Perona and J. Malik, "Scale-space and edge detection using anistropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 9, pp. 629–639, July 1990.

[6] C. M. Stein, "Estimation of the mean of a multivariate normal distribution," *The Annals of Statistics*, vol. 9, no. 6, pp. 1135–1151, November 1981.

[7] D. Van De Ville M. Kocher, "Nonlocal means with dimensionality reduction and sure-based parameter selection," *IEEE Transactions on Image Processing*, vol. 20, no. 9, pp. 2683–2690, September 2011.

[8] H. Talebi and P. Milanfar, "Patch-wise ideal stopping time for anisotropic diffusion," *SPIE Conference on Visual Information Processing and Communication*, January 2012.

[9] P. Chatterjee and P. Milanfar, "Patch-based near-optimal denoising," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1635–1649, April 2012.