Bayesian Inference for Neighborhood Filters with Application in Denoising

Chao-Tsung Huang
National Tsing Hua University, Taiwan
chaotsung@ee.nthu.edu.tw

Abstract

Range-weighted neighborhood filters are useful and popular for their edge-preserving property and simplicity, but they are originally proposed as intuitive tools. Previous works needed to connect them to other tools or models for indirect property reasoning or parameter estimation. In this paper, we introduce a unified empirical Bayesian framework to do both directly. A neighborhood noise model is proposed to reason and infer the Yaroslavsky, bilateral, and modified non-local means filters. An EM+ algorithm is devised to estimate the essential parameter, range variance, via the model fitting to empirical distributions. Finally, we apply this framework to color-image denoising. Experimental results show that the proposed model fits noisy images well and the range variance is estimated successfully. The image quality can also be improved by a proposed recursive fitting and filtering scheme.

1. Introduction

Range-weighted formulation has been widely used to provide edge-preserved denoising since the introduction of local neighborhood filtering, especially the Yaroslavsky [28] and bilateral [23] filters. Many variations were also proposed for different applications, such as the trilateral filter for high contrast images[6] and the cross bilateral filter for fusing image pairs [20]. The reader is referred to [17] for more applications. The non-local means (NLM) [2] was further proposed for a non-local neighborhood and got fruitful patch-based extensions on denoising.

The intuitive idea behind is to assign filter coefficients based on similarity, e.g. the bilateral filter is given by

$$\hat{\mathbf{z}}_l = \frac{\sum_{i\in\Lambda_l} w_{l,i} d_{l,i} \cdot \mathbf{y}_i}{\sum_{i\in\Lambda_l} w_{l,i} d_{l,i}} \tag{1}$$

where $\mathbf{y}$ and $\hat{\mathbf{z}}$ are the observed and filtered signals respectively, and Λ_l represents the neighborhood at position l. The adaptive kernel consists of one range-weighted $w_{l,i} = \exp(-\frac{\|\mathbf{y}_l - \mathbf{y}_i\|_2^2}{2\sigma_r^2})$ and one distance-weighted $d_{l,i} = \exp(-\frac{\|l-i\|_2^2}{2\sigma_d^2})$. The former adapts to pixel similarity for edge preservation, which is controlled by range variance σ_r^2. And the latter provides a spatial smoothing window. If the spatial weights are all equal, it will degenerate to the Yaroslavsky filter. On the other hand, it will evolve to the NLM if $w_{l,i}$ is defined by patch similarity:

$$w_{l,i} = \exp(-\frac{\sum_{b\in\mathcal{B}} \| \mathbf{y}_{l+b} - \mathbf{y}_{i+b} \|_2^2}{2B\sigma_r^2}),\ B = |\mathcal{B}|, \tag{2}$$

where $\{\mathbf{y}_{i+b} | b \in \mathcal{B}\}$ forms the patch at position i.

Filtering based on range-weighted similarity is effective but lacks of theoretical basis. For understanding its mathematical properties, several previous works studied their connections to other classical methods, such as mean shift [1], anisotropic diffusion [3] and Bayesian approach [8, 16], and thus found improvements on the neighborhood filters. However, these connections were unable to provide further information to infer the range variance σ_r^2 directly from the observed data.

In contrast, without reasoning the properties statistical techniques have been adopted to estimate the parameters indirectly using the basic observation model

$$\mathbf{y} = \mathbf{z} + \mathbf{n}, \tag{3}$$

where $\mathbf{n}$ is additive Gaussian noise. The χ^2 test was used to choose the parameters for the NLM filter in [11]. The Stein's unbiased risk estimate (SURE) [22] can provide unbiased estimation of the mean squared error (MSE) from noisy and filtered images. Thus many parameter combinations can be tested, and the one giving the smallest estimated MSE can be selected. Though accurate, the complexity is quite high because each combination needs to filter the whole image separately.

Contribution. In this paper, we build a unified empirical Bayesian framework to both infer the neighborhood filters and estimate their range variance. We first introduce a novel neighborhood noise model. It reasons the range-weighted formulation using maximum a posteriori (MAP) estimation via a soft-edge prior and infers the filters using maximum likelihood (ML) estimation on different likelihood

functions. We then present an EM+ algorithm to perform the model fitting for estimating the range variance σ_r^2 from noisy images. From the experimental results on color-image denoising, we show that the proposed model fits noisy images well and estimates σ_r^2 as accurate as SURE does. The advantage over SURE is also shown both computation-wise and quality-wise when considering a proposed recursive filtering scheme. We also believe that this framework can be extended to other range-weighted algorithms for deeper theoretical understanding and better parameter estimation.

2. Related Work

Joint filter/parameter inference. In [21], a wavelet-domain denoising algorithm was developed with the assumption that the latent image $\mathbf{z}$ is Gaussian scale mixtures (GSM) [26]. The adaptive parameters can then be statistically inferred from the neighborhood. If detailed camera information is available, [13] can perform noise estimation and removal automatically. Not surprisingly, these filters are different from the range-weighted ones. The PLOW filter [4] is equivalent to the NLM filter plus a residual filter. Although the residual filter is based on the covariance matrices inferred from geometric clusters, the range variance was still given in a heuristic way. In this paper, we target the joint inference for neighborhood filters.

SURE-based parameter estimation. The SURE-based method gives the state-of-the-art accuracy for parameter estimation, and it is basically applicable to any parameter. It needs to estimate the noise variance first, and one popular method is by median absolute deviation (MAD). It has been applied to the bilateral filter for grey [18] and multispectral images [19] and also to the NLM filter [24, 25, 5]. A fast implementation for the bilateral filter was also proposed in [12], but several passes of filtering were still required. For the SURE-based method to work well, two conditions should be met: the Gaussian noise assumption and a differentiable filter kernel on the noisy image $\mathbf{y}$. In this paper, the model fitting itself is only able to estimate the range variance. However, we will show its applicability to two cases in which the SURE-based method would fail. One is recursive filtering for which the noise is no longer Gaussian. The other one is for the NLM filters which use motion estimation to select candidate patches such that the kernel is dynamic and thus indifferentiable on $\mathbf{y}$.

Image denoising. Several types of approaches have been studied, including local-based [28, 23], transform-based [21, 27], nonlocal-based [2, 4], and sparsity-based (e.g. BM3D [7], NLSM[14] and WNNM[9]). The last one showed superior image quality recently, which was based on grouping with patch similarity and optimization for sparse representation. However, the parameters are mostly given heuristically. Besides, they were often proposed and optimized for grey images, and additional modification is required to support color channels. In contrast, our method can work on multi-channel signals directly.

3. Noise Model and Bayesian Inference

We will first propose a novel noise model and infer the Yaroslavsky filter directly from it. By modifying the likelihood function to improve the robustness of estimation, we will then infer the bilateral filter and a modified NLM filter.

3.1. Neighborhood noise model (NNM)

For a latent k-channel signal $\mathbf{z}_l$ at position l, we formulate its neighbors $\mathbf{y}_{i\in\Lambda_l}$ by GSM to model localized intensity edges using soft decisions:

$$\mathbf{y}_i = \mathbf{z}_l + \frac{\mathbf{n}_{l,i}}{\sqrt{w_{l,i}}}, \tag{4}$$

where $\mathbf{n}_{l,i}$ are additive white Gaussian noises (AWGN) of covariance matrix $\sigma^2\mathbf{I_k}$ and $w_{l,l} = 1$ as the basic model. For the neighboring pixels, a smaller $w_{l,i}$ means a more widely distributed $\mathbf{y}_i$, so there is more likely an edge between positions l and i. Otherwise, smooth texture will be inferred if $w_{l,i}$ is close to one. To infer the Gaussian range-weighted kernel in (1), $w_{l,i\neq l}$ are defined as white hidden variables of a prior with two parameters ϵ and α:

$$f_w(w;\epsilon,\alpha) = \frac{1}{N(\epsilon,\alpha)} w^{-k/2} w^{-\alpha w} e^{\alpha w},\ w\in[\epsilon,1], \tag{5}$$

where $N(\epsilon,\alpha) = \int_\epsilon^1 w^{-k/2} w^{-\alpha w} e^{\alpha w}\, dw$ for normalization, α will link σ^2 to σ_r^2, and non-zero ϵ can guarantee the convergence of the integration for $k \geq 3$.

3.2. Inference for Yaroslavsky filter

Given observed data $\mathbf{y}_i$, we have the posterior $\Phi_l \propto p(\mathbf{y}_l;\mathbf{z}_l)\prod_{i\in\Lambda_l, i\neq l} p(\mathbf{y}_i|w_{l,i};\mathbf{z}_l) f_w(w_{l,i})$. Its energy function L_l can then be derived:

$$L_l = \frac{g_{l,l}^2}{2\sigma^2} + \sum_{i\neq l}\frac{w_{l,i} g_{l,i}^2}{2\sigma^2} - \log w_{l,i}^{\frac{k}{2}} - \log f_w(w_{l,i}), \tag{6}$$

where $g_{l,i}^2 \triangleq \| \mathbf{z}_l - \mathbf{y}_i \|_2^2$. The estimation for $w_{l,i}$ and $\mathbf{z}_l$ can be derived by minimizing L_l:

$$\frac{\partial L_l}{\partial w_{l,i}} = 0 \quad \Rightarrow w_{l,i} = e^{-\frac{g_{l,i}^2}{2\sigma_r^2}},\ \sigma_r^2 = \alpha\sigma^2,\ (i\neq l) \tag{7}$$

$$\frac{\partial L_l}{\partial \mathbf{z}_l} = 0 \quad \Rightarrow \mathbf{z}_l = \frac{\sum_{i\in\Lambda_l} w_{l,i}\cdot\mathbf{y}_i}{\sum_{i\in\Lambda_l} w_{l,i}}. \tag{8}$$

The Yaroslavsky filter is then equivalent to the first-iteration estimation for solving this fixed-point problem with an initial condition $\mathbf{z}_l^{(0)} = \mathbf{y}_l$.

The NNM estimation (7) and (8) can be further interpreted into two sequential steps respectively:

1. Independent MAP estimation for each $w_{l,i}$ by maximizing $p(\mathbf{y}_i|w_{l,i};\mathbf{z}_l)f_w(w_{l,i})$ when given a fixed $\mathbf{z}_l$;
2. ML estimation for $\mathbf{z}_l$ by optimizing the likelihood $\mathcal{L}_z(\mathbf{z}_l) = \frac{g_{l,l}^2}{2\sigma^2} + \sum_{i\neq l}\frac{w_{l,i}g_{l,i}^2}{2\sigma^2}$ when given fixed $w_{l,i}$.

By modifying the likelihood function in the second step for considering proximity and patch similarity, the bilateral and NLM filters can be derived respectively as follows.

3.3. Extension for bilateral filter

To consider proximity in a nonparametric way using the Gaussian weighting kernel $d_{l,i}$, the locally weighted maximum likelihood (LWML) [15] can be applied to the likelihood $\mathcal{L}_z$ to have the pseudo likelihood function

$$\tilde{\mathcal{L}}_z(\mathbf{z}_l) = d_{l,l}\cdot\frac{g_{l,l}^2}{2\sigma^2} + \sum_{i\neq l} d_{l,i}\cdot\frac{w_{l,i}g_{l,i}^2}{2\sigma^2}. \tag{9}$$

Then the LWML estimation by $\frac{\partial\tilde{\mathcal{L}}_z}{\partial\mathbf{z}_l} = 0$ gives the same formulation as the bilateral filter.

3.4. Extension for NLM filter

The corresponding NNM for each latent pixel in a patch $\{\mathbf{z}_{l+b},\, b\in\mathcal{B}\}$ and its neighborhood Λ_{l+b} is as defined in (4). The first-iteration MAP estimation for each soft-edge variable then gives

$$w_{l+b,i+b} = e^{-\frac{\|\mathbf{y}_{l+b}-\mathbf{y}_{i+b}\|_2^2}{2\sigma_r^2}},\ i\in\Lambda_l. \tag{10}$$

Let the column vectors of the latent patch and the observed patches be $\mathbf{Z}_l$ and $\mathbf{Y}_{i\in\Lambda}$. They are formed by cascading $\mathbf{z}_{l+b}$ and $\mathbf{y}_{i+b}$ respectively in a predefined order Υ for $b\in\mathcal{B}$. Then the partial likelihood from $\mathbf{Y}_i$ becomes

$$\mathcal{L}(\mathbf{Z}_l;\mathbf{Y}_i) = G(\mathbf{Z}_l;\mathbf{Y}_i,\mathbf{\Sigma}_{l,i}), \tag{11}$$

where $G(\cdot;\mu,\mathbf{\Sigma})$ is a multivariate Gaussian function, and the diagonal entries of $\mathbf{\Sigma}_{l,i}$ are formed by cascading $\mathrm{diag}(\frac{\sigma^2}{w_{l+b,i+b}}\mathbf{I}_k)$ in the predefined order Υ. The entries outside of the diagonal have no effect on the following derivation.

For NLM filtering, one observed patch $\mathbf{Y}_i$ has only one summation weight $W_{l,i}$. Thus we use a patch-based likelihood function to approximate (11), which is devised as

$$\tilde{\mathcal{L}}(\mathbf{Z}_l;\mathbf{Y}_i) = G(\mathbf{Z}_l;\mathbf{Y}_i,\frac{\sigma^2}{W_{l,i}}\mathbf{I}_{kB}). \tag{12}$$

Since these two likelihood functions both behave like probability density functions (pdf), we choose $W_{l,i}$ by minimizing the Kullback-Leibler divergence (KLD) between them:

$$\mathcal{D}_{NLM} \triangleq \mathcal{D}_{KL}(\mathcal{L}\parallel\tilde{\mathcal{L}}) = -\int\mathcal{L}\log\frac{\tilde{\mathcal{L}}}{\mathcal{L}}\,d\mathbf{Z}_l, \tag{13}$$

$$\begin{aligned}\frac{\partial\mathcal{D}_{NLM}}{\partial W_{l,i}} &= -\int\mathcal{L}\left(\frac{kB}{2}\frac{1}{W_{l,i}} - \frac{\|\,\mathbf{Z}_l-\mathbf{Y}_i\,\|_2^2}{2\sigma^2}\right)d\mathbf{Z}_l \\ &= -\left(\frac{kB}{2}\frac{1}{W_{l,i}} - \frac{1}{2}\sum_{b\in\mathcal{B}}\frac{k}{w_{l+b,i+b}}\right) = 0, \end{aligned}\tag{14}$$

$$\Rightarrow W_{l,i} = \left(\sum_{b\in\mathcal{B}} w_{l+b,i+b}^{-1}/B\right)^{-1}. \tag{15}$$

Then the ML estimation for the combined patch-based likelihood functions $\sum_{i\in\Lambda}\tilde{\mathcal{L}}(\mathbf{Z}_l;\mathbf{Y}_i)$ becomes

$$\mathbf{Z}_l = \frac{\sum_{i\in\Lambda_l} W_{l,i}\cdot\mathbf{Y}_i}{\sum_{i\in\Lambda_l} W_{l,i}}, \tag{16}$$

which suggests a modified NLM (MNLM) filter with the coefficients $W_{l,i}$ in (15). Note that $W_{l,i}$ is the harmonic average of $w_{l+b,i+b}$, while the conventional NLM uses the geometric average as shown in (2).

4. Model Fitting and Parameter Estimation

In the following, we will first introduce how to build a robust observation model in chi scale mixtures and then how we fit the model using an expectation-maximization-plus (EM+) algorithm.

4.1. Observable chi scale mixtures (CSM)

Let $s_{l,i} \triangleq \|\,\mathbf{y_l} - \mathbf{y_i}\,\|_2$. It is independent of $\mathbf{z_l}$ and with the CSM formulation:

$$s_{l,i} = \sigma\sqrt{\frac{w_{l,i}+1}{w_{l,i}}}u_{l,i}, u_{l,i}\sim\chi_k, \tag{17}$$

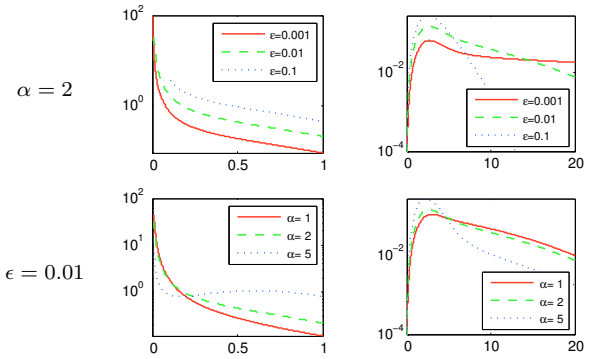
where $u_{l,i}$ is a chi distribution with k degrees of freedom. For simplicity, we use $s = s_{l,i}$ and $w = w_{l,i}$ in the following. The marginal pdf of s can be derived by

$$f_s(s;\sigma,\epsilon,\alpha) = \int_\epsilon^1 f_{s,w}(s,w;\sigma,\epsilon,\alpha)\,dw, \tag{18}$$

$$f_{s,w} = \frac{1}{T(\epsilon,\alpha)}\frac{s^{k-1}\sigma^{-k}}{(w+1)^{\frac{k}{2}}}e^{-\frac{w}{w+1}\frac{s^2}{2\sigma^2}}w^{-\alpha w}e^{\alpha w}, \tag{19}$$

where $f_{s,w}$ is the joint pdf of s and w, and $T(\epsilon,\alpha) = 2^{\frac{k}{2}-1}\Gamma(k/2)N(\epsilon,\alpha)$ for normalization.

Fig. 1 shows how the soft-edge prior $f_w(w)$ and the CSM pdf $f_s(s)$ behave with different α and ϵ. $f_w(w)$ represents how likely intensity edges appear, i.e. more small w for more edges. For smaller α or ϵ, $f_w(w)$ tilts more to the left and $f_s(s)$ has a longer tail such that more edges are expected. For larger α or ϵ, $f_w(w)$ concentrates more on the right, and $f_s(s)$ gets closer to a chi distribution. Thus we can model noisy images by varying α and ϵ to fit different image properties and varying σ for different noise intensity.



(a) $f_w(w)$ (b) $f_s(s), \sigma = 1$

Figure 1. Distributions of the soft-edge w and the observable chi scale mixtures s with different values for α and ϵ ($k = 3$).

Another advantage to define s in l^2-norm is its robustness. When the components of $\mathbf{y}_i$ are not independent in the observed color space, e.g. RGB, we may apply an orthogonal transform to diagonalize their covariance matrix $\Sigma_{\mathbf{y}}$ for decorrelation. However, since the l^2-norm is invariant to the orthogonal transform, we can calculate s (and w) in the observed color space without performing the decorrelation. Thus the model fitting (and the filtering) can be applied on the observed $\mathbf{y}_i$ directly without loss of optimality.

4.2. EM+ algorithm for CSM fitting

Given an observed data set $\mathcal{S}$, its empirical distribution is defined by $P(s \in \mathcal{S})$. Then we estimate the corresponding pdf $f_s(s)$ by iteratively updating the model parameters based on $P(s)$. Assume in the previous iteration the estimated parameters are σ, α, and ϵ. Our EM+ algorithm updates them to $\tilde{\sigma}$, $\tilde{\alpha}$, and $\tilde{\epsilon}$ through the following three steps: EM update, KLD update, and Quasi-Newton (QN) update.

4.2.1 EM update: $(\sigma, \alpha, \epsilon) \Rightarrow (\hat{\sigma}, \hat{\alpha}, \epsilon)$

For simplicity, we will ignore the σ, α, and ϵ in $f_{s,w}$ and f_s if the parameters in the previous iteration are used. The expected value of the log likelihood function can be derived as $Q(\hat{\sigma}, \hat{\alpha}|\sigma, \alpha) = \sum_j P(s_j) q(s_j, \hat{\sigma}, \hat{\alpha}|\sigma, \alpha)$ where

$$q(s, \hat{\sigma}, \hat{\alpha}|\sigma, \alpha) = \int_{\epsilon}^{1} p(w|s) \mathcal{L}_{EM}(\hat{\sigma}, \hat{\alpha}, \epsilon; w, s)\, dw, \quad (20)$$

$$p(w|s) = f_{s,w}/f_s, \; \mathcal{L}_{EM} = \log f_{s,w}(s, w; \hat{\sigma}, \epsilon, \hat{\alpha}). \quad (21)$$

Set $\frac{\partial Q}{\partial \hat{\sigma}} = \frac{\partial Q}{\partial \hat{\alpha}} = 0$ to have the EM update for $\hat{\sigma}$ and $\hat{\alpha}$:

$$\hat{\sigma}^2 = \frac{1}{k} \sum_j P(s_j) \cdot \frac{\int_{\epsilon}^{1} f_{s,w}(s_j, w) s_j^2 \frac{w}{w+1}\, dw}{f_s(s_j)}, \quad (22)$$

$$H(\hat{\alpha}, \epsilon) = \sum_j P(s_j) \cdot \frac{\int_{\epsilon}^{1} f_{s,w}(s_j, w) w(1 - \log w)\, dw}{f_s(s_j)}, \quad (23)$$

where $H(\hat{\alpha}, \epsilon) \triangleq \mathbf{E}_{w;\hat{\alpha},\epsilon}[w(1 - \log w)]$ and it is an increasing function with respect to $\hat{\alpha}$ because $\partial H/\partial \hat{\alpha} = \mathbf{Var}_{w;\hat{\alpha},\epsilon}[w(1 - \log w)] \geqq 0$. Thus the corresponding $\hat{\alpha}$ can be found quickly using a bisection search on $H(\hat{\alpha}, \epsilon)$.

4.2.2 KLD update: $(\hat{\sigma}, \hat{\alpha}, \epsilon) \Rightarrow (\hat{\sigma}, \hat{\alpha}, \hat{\epsilon})$

The EM algorithm is unable to update ϵ because the support of w for $p(w|s)$ and $\mathcal{L}_{EM}$ in (20) should be the same. Instead, we update it by optimizing the approximate KLD $\mathcal{D} = \sum_j -P(s_j) \cdot \log \frac{f_s(s_j;\hat{\sigma},\epsilon,\hat{\alpha})}{P(s_j)}$. With $\frac{\partial \mathcal{D}}{\partial \epsilon} = 0$, a fixed-point representation of the optimal ϵ can be derived. And we use the first-iteration result with an initial condition $\hat{\epsilon}^{(0)} = \epsilon$ as our updating formulation:

$$\left(\frac{\hat{\epsilon}+1}{\hat{\epsilon}}\right)^{\frac{k}{2}} = \sum_j P(s_j) \cdot \frac{s_j^{k-1} \hat{\sigma}^{-k} e^{-\frac{\epsilon}{\epsilon+1}\frac{s_j^2}{2\hat{\sigma}^2}}}{2^{\frac{k}{2}-1}\Gamma(\frac{k}{2}) f_s(s_j; \hat{\sigma}, \epsilon, \hat{\alpha})}. \quad (24)$$

4.2.3 QN update: $(\hat{\sigma}, \hat{\alpha}, \hat{\epsilon}) \Rightarrow (\tilde{\sigma}, \tilde{\alpha}, \tilde{\epsilon})$

The update formulations in (22), (23), and (24), require numerical evaluations for lots of integrals, which prolongs the execution time for one iteration. Besides, we found that the update step sizes are usually small near the optimizing point, which increases the number of iterations. Thus the above updates take a long time to converge in many cases. We adopt the QN method, QN1 in [10], to accelerate the fitting process. Let $\boldsymbol{\theta} = (\sigma, \alpha, \epsilon)^T$, $\hat{\boldsymbol{\theta}} = (\hat{\sigma}, \hat{\alpha}, \hat{\epsilon})^T$, $\tilde{\boldsymbol{\theta}} = (\tilde{\sigma}, \tilde{\alpha}, \tilde{\epsilon})^T$, and $\mathbf{F}(\boldsymbol{\theta}) = \hat{\boldsymbol{\theta}}(\boldsymbol{\theta}) - \boldsymbol{\theta}$. The QN1 method solves $\mathbf{F}(\boldsymbol{\theta}) = \mathbf{0}$ by maintaining a matrix $\mathbf{A}$ which approximates the inverse Jacobian matrix $\mathbf{J}_{\mathbf{F}}^{-1}$ and is updated using the Broyden's update method. Then the QN update is derived by $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta} + \boldsymbol{\delta}$ where

$$\boldsymbol{\delta} = -\mathbf{A} \cdot \mathbf{F}(\boldsymbol{\theta}). \quad (25)$$

However, in practice the QN update could be unstable when $\mathbf{A}$ has a high condition number. Some heuristic, e.g. reinitializing $\mathbf{A}$, is required in this situation. We choose to confine $\tilde{\boldsymbol{\theta}}$ in a reasonably large range $\boldsymbol{\Psi}$ for its quick convergence. When $\tilde{\boldsymbol{\theta}}$ is out of $\boldsymbol{\Psi}$, we will directly set $\boldsymbol{\delta}$ to $\mathbf{F}(\boldsymbol{\theta})$ as the original EM/KLD update and may further scale down its value to make $\tilde{\boldsymbol{\theta}}$ inside $\boldsymbol{\Psi}$ if necessary.

4.3. Discriminative capability of KLD

The fitting quality relies on the discriminative capability of KLD between different parameters. Fig. 2 shows the KLD discrimination of $f_s(s)$ for $k = 3$ when α has a small change $\triangle\alpha = 0.5$ and when ϵ increases by $\triangle\epsilon = 0.005$. It is clear that the discriminative capability declines as α or ϵ increases. For sufficiently large α or ϵ, $f_s(s)$ is very similar to a chi distribution and KLD becomes insensitive to the parameters. Therefore, some upper bounds for α and ϵ may be chosen without compromising KLD.

5. Application in Image Denoising

For the inferred filters, each neighborhood can be viewed as a realization of the NNM. Thus the CSM model fitting can provide parameter estimation. In the following, we will apply this approach to color-image denoising ($k = 3$).

5.1. Parameter estimation

Before model fitting, we need to derive the empirical distribution $P(s)$. For the Yaroslavsky filter, we have a fixed neighborhood Λ_l for each pixel, and $P(s)$ can be simply obtained by accumulating the histogram of all $s_{l,i}$. For the bilateral filter, we construct its $P(s)$ by accumulating the histogram with the spatial weight $d_{l,i}$ to consider proximity. For example, the frequency of $s = 2$ will be increased by $d_{l,i}$ for an event $s_{l,i} = 2$. For the MNLM filter, we consider a dynamic neighborhood Λ_l which may depend on patch similarity sorting or hard thresholding for better performance. In this case, we need to do the computation to derive Λ_l and then accumulate the histogram of $s_{l+b,i+b}$.
Given $P(s)$, we perform the CSM fitting using the EM+ algorithm and select the final result based on the KLD calculated for $P(s)$ and the estimated distribution $\tilde{P}(s;\boldsymbol{\theta}^{(m)})$ in each iteration. Besides, we also devise an ϵ-bounded estimation to handle the KLD insensitivity issue. It is activated when ϵ is close to a given upper bound ϵ_{bd}, i.e. $|\epsilon - \epsilon_{bd}| < \triangle\epsilon$. Since the sensitivity of KLD becomes small when ϵ is near ϵ_{bd}, we can directly compare KLD through $\epsilon = \epsilon_{bd}$ using a bisection search on α. For each α, the σ-update (22) which converges very quickly is used to find the best corresponding σ and KLD.
The algorithm parameters which are fixed in this paper are as follows: maximum fitting iteration $M = 15$, initial CSM parameter $\boldsymbol{\theta}_{ini} = (\frac{s_{md}}{2}, 3, 10^{-3})$ where s_{md} is the mode of $P(s)$, ϵ-bound criteria $\triangle\epsilon = 10^{-3}$, and parameter range $\boldsymbol{\Psi} = \{\boldsymbol{\theta} | \sigma \geqq 10^{-5}, \alpha \in [3, 15], \epsilon \in [10^{-5}, \epsilon_{bd}]\}$. The convergence condition is that the KLD is smaller than 10^{-5} or σ_r^2 changes by less than 0.1%. Note that $\boldsymbol{\Psi}$ is used to avoid unstable QN updates and is defined sufficiently large such that only seldom final results touch the boundaries. The only exception is ϵ_{bd} which can be used to activate the ϵ-bounded estimation.

5.2. Recursive local filter

It is shown that the conventional Yaroslavsky/bilateral filter is equivalent to the first-iteration MAP/ML estimation. Simply applying more iterations with the same σ_r^2 indeed reduces the energy function (6) but does not help for increasing PSNR. Instead, we propose to apply the model fitting and filtering recursively. Each iteration estimates the NNM parameters for the current noisy image $\hat{\mathbf{y}}^{(m)}$ and performs filtering on it with the estimated range variance $\hat{\sigma}_r^{2(m)}$. This process is terminated when the filter iteration exceeds M_{flt} times or the estimated noise intensity $\hat{\sigma}^{(m)}$ is smaller than a threshold σ_{cl} which represents a clean image.

5.3. Recursive MNLM filter

The proposed recursive MNLM filter has three differences from the recursive local one. First, the basic processing unit becomes a patch, instead of a pixel, and a flag b_{ag} decides how to aggregate these patches into one image. If $b_{ag} = 1$, each image pixel at position l will be derived by averaging all its corresponding values in neighboring estimated patches $\hat{\mathbf{X}}_{l+b}, b \in \mathcal{B}$. Otherwise, no aggregation will be performed. Second, the neighborhood for each patch is constructed dynamically based on some given constraints. Third, a DCT-Wiener filter is introduced to increase the performance, which is activated by a flag b_{dct}.
The DCT-Wiener filter serves similarly as the residual filter in PLOW. We use the DCT, denoted as $\mathcal{T}(\cdot)$, to approximate the decorrelation matrix and then apply element-wise Wiener filtering to update the patch $\hat{\mathbf{X}}_l$

$$\hat{\mathbf{X}}_l \leftarrow \mathcal{T}^{-1}(\mathbf{W}_{wie} \circ \mathcal{T}(\hat{\mathbf{X}}_l)), \tag{26}$$

where the element of the shrinkage matrix $\mathbf{W}_{wie}$ is

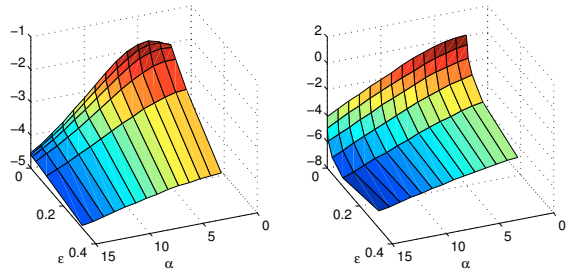
$$(\mathbf{W}_{wie})_{i'j'k'} = \frac{\hat{\sigma}^2_{X,i'j'k'}}{\hat{\sigma}^2_{X,i'j'k'} + \hat{\sigma}^2_l}, \tag{27}$$

and the signal variance is estimated from neighbors by $\hat{\sigma}^2_{X,i'j'k'} = \mathbf{E}_{i\in\Lambda_l}[(\mathcal{T}(\hat{\mathbf{Y}}_i^{(m)}))^2_{i'j'k'}] - \hat{\sigma}^{2(m)}$ and the noise variance by $\hat{\sigma}^2_l = \frac{\hat{\sigma}^{2(m)}}{\sum_{i\in\Lambda_l} W_{l,i}}$ due to weighted averaging. The recursive MNLM filter is summarized in Algorithm 1.

Algorithm 1 Recursive MNLM Filter

Input: Noisy image $\mathbf{y}$; ϵ-bound ϵ_{bd}; DCT-Wiener flag b_{dct}; Aggregation flag b_{ag}
Output: Denoised image $\hat{\mathbf{z}}$

Initialize $\hat{\mathbf{y}}^{(1)} = \mathbf{y}$, $\hat{\mathbf{z}} = \mathbf{y}$, $m = 1$
repeat
  Get empirical $P(s)$ of $\hat{\mathbf{y}}^{(m)}$ by constructing each Λ_l
  Get $\hat{\sigma}_r^{(m)}$, $\hat{\sigma}^{(m)}$ using parameter estimation
  if $(\hat{\sigma}^{(m)} \geq \sigma_{cl})$ **then**
    for each patch $\mathbf{Y}_l$ in $\hat{\mathbf{y}}^{(m)}$ **do**
      Get the estimation $\hat{\mathbf{X}}_l = \frac{\sum_{i\in\Lambda_l} W_{l,i}\cdot\hat{\mathbf{Y}}_i^{(m)}}{\sum_{i\in\Lambda_l} W_{l,i}}$
      Perform DCT-Weiner filtering with b_{dct}
    end for
    Aggregate $\hat{\mathbf{X}}_l$ to form $\hat{\mathbf{z}}^{(m)}$ with b_{ag}
    $\hat{\mathbf{y}}^{(m+1)} \leftarrow \hat{\mathbf{z}}^{(m)}$, $\hat{\mathbf{z}} \leftarrow \hat{\mathbf{z}}^{(m)}$
  end if
  $m \leftarrow m + 1$
until $(m > M_{flt})$ or $(\hat{\sigma}^{(m)} < \sigma_{cl})$



(a) $\log_{10} \mathcal{D}_{\triangle\alpha=0.5}$ (b) $\log_{10} \mathcal{D}_{\triangle\epsilon=0.005}$

Figure 2. KLD discrimination ($k = 3$) on (a) α and (b) ϵ, where $\mathcal{D}_{\triangle\alpha}(\alpha, \epsilon) \triangleq \mathcal{D}_{KL}(f_s(s;\sigma,\epsilon,\alpha) \parallel f_s(s;\sigma,\epsilon,\alpha+\triangle\alpha))$ and $\mathcal{D}_{\triangle\epsilon}(\alpha, \epsilon) \triangleq \mathcal{D}_{KL}(f_s(s;\sigma,\epsilon,\alpha) \parallel f_s(s;\sigma,\epsilon+\triangle\epsilon,\alpha))$. Note that σ affects only the scaling and has no effect on $\mathcal{D}_{\triangle\alpha}$ and $\mathcal{D}_{\triangle\epsilon}$.

6. Experiments on Model Fitting and Filtering

6.1. Parameter setting

Six filter configurations are tested:

1. **YF-5×5**: Yaroslavsky filter, Λ: 5×5, $\epsilon_{bd} = 0.1/1.0$;
2. **YF-9×9**: Yaroslavsky filter, Λ: 9×9, $\epsilon_{bd} = 0.1/1.0$;
3. **BF-9×9**: Bilateral filter, Λ: 9×9, $\epsilon_{bd} = 0.1/1.0$;
4. **BF-13×13**: Bilateral filter, Λ: 13×13, $\epsilon_{bd} = 0.1/1.0$;
5. **MNLM-Simple**: MNLM filter, $\epsilon_{bd} = 0.1$, $b_{dct} = 0$ (DCT-Wiener off), $b_{ag} = 0$ (no aggregation);
6. **MNLM-DCT**: MNLM filter, $\epsilon_{bd} = 1.0$ (ϵ-bound off), $b_{dct} = 1$ (DCT-Wiener on), $b_{ag} = 1$ (one-pixel grid).

Two settings of ϵ_{bd} are tested for local filters. The basic $\epsilon_{bd} = 1.0$ simply follows the definition in (5). In contrast, the empirical $\epsilon_{bd} = 0.1$ will use the ϵ-bounded estimation for large ϵ. The spatial weight of bilateral filters σ_d is set to the radius of Λ, e.g. $\sigma_d = 4$ for $\Lambda = 9\times9$. The patch size for the MNLM filter is 9×9. For constructing MNLM neighborhood, we apply motion estimation in a 31×31 search window around position l and choose the best ten candidates as Λ_l. For filter termination, the maximum iteration number M_{flt} is set to 3 and σ^2_{cl} set to 10.

To obtain the best σ_r^2 and the SURE estimation for comparison, we perform a σ_r^2 scan (30 values) from $\alpha = 0.5$ to 25.0 for each test condition. The SURE-based method uses the noise variance σ^2_{MAD} estimated by MAD as done in [4] and is denoted as MAD+SURE. Twelve standard color images of different properties are used in our experiments with AWGN of five different intensity values of σ_n.

6.2. Yaroslavsky/Bilateral filter

Fig. 3 shows one typical example of the test results. The results for other test filters and test images can be browsed online[1]. The CSM model can fit the long-tailed empirical distributions well no matter when the noise is small or large. It also successfully predicts that α should become larger as the noise intensity σ_n increases, while the conventional heuristics usually apply a fixed value.

Table 1 lists the result of **BF-9×9** ($\epsilon_{bd} = 0.1$) in detail. MAD+SURE fails in two cases. One is for small σ_n (e.g. σ_n=5, 1st iteration) due to the inaccuracy of σ^2_{MAD}. The other one is for iterations after the first iteration (e.g. σ_n=50, 2nd iteration) because the noise is not Gaussian any more. In contrast, the CSM estimation performs well in terms of both PSNR and σ_r^2 accuracy in these two cases. It means that the edge information can be well captured by the CSM model even when the noise is small or becomes non-Gaussian. This useful property also enables the proposed recursive scheme. Besides, in other cases the CSM estimation shows similar performance compared to the MAD+SURE. An interesting property can also be found. The CSM estimation usually underestimates the noise variance σ^2, which means it may mistake noises for edges. In contrast, the MAD+SURE tends to overestimate because the MAD may mistake edges for noises.

The execution time is also shown in Table 1. The proposed method runs much faster than the MAD+SURE since it does not require a σ_r^2 scan. Each iteration of it consists of three steps: getting $P(s)$, CSM fitting, and filtering. The first and third steps take time proportionally to the image resolution and filter support size (7.9 s and 6.5 s on average respectively). In contrast, the fitting time depends on the EM+ iteration numbers and whether the ϵ-bounded estimation is turned on. For larger noises, the fitting tends to be insensitive to KLD, and thus more time is required, e.g. 9.6 s on average for $\sigma_n = 50$ (1st iteration) while only 3.5 s for $\sigma_n = 5$. The percentage of the fitting time will be smaller for larger images. Note that MAD+SURE not only fails to predict in the recursive iterations but also requires significantly higher computation complexity due to the σ_r^2 scan.

The test results of Yaroslavsky and bilateral filters are summarized in Table 2. For $\epsilon_{bd} = 0.1$, the CSM estimation performs comparably to the SURE in the first iteration for large σ_n and outperforms for small σ_n. Moreover, the proposed recursive fitting and filtering can increase PSNR by up to 1.2 dB. As for $\epsilon_{bd} = 1.0$, its first-iteration results are not as good as $\epsilon_{bd} = 0.1$ for its less accurate σ_r^2 estimation and higher KLD (due to KLD insensitivity). However, the recursive filtering can recover its quality drop and even make it slightly better than $\epsilon_{bd} = 0.1$ with a little more iterations. Therefore, the basic $\epsilon_{bd} = 1.0$ gives slightly better quality when using recursive filtering, but the empirical $\epsilon_{bd} = 0.1$ may be preferred when only one or fewer iterations are allowed. The choice of σ_r^2 affects the performance much more sensitively than that of Λ and $d_{l,i}$, which also justifies the need of a good σ_r^2 estimator.

[1] http://www.ee.nthu.edu.tw/chaotsung/nnm

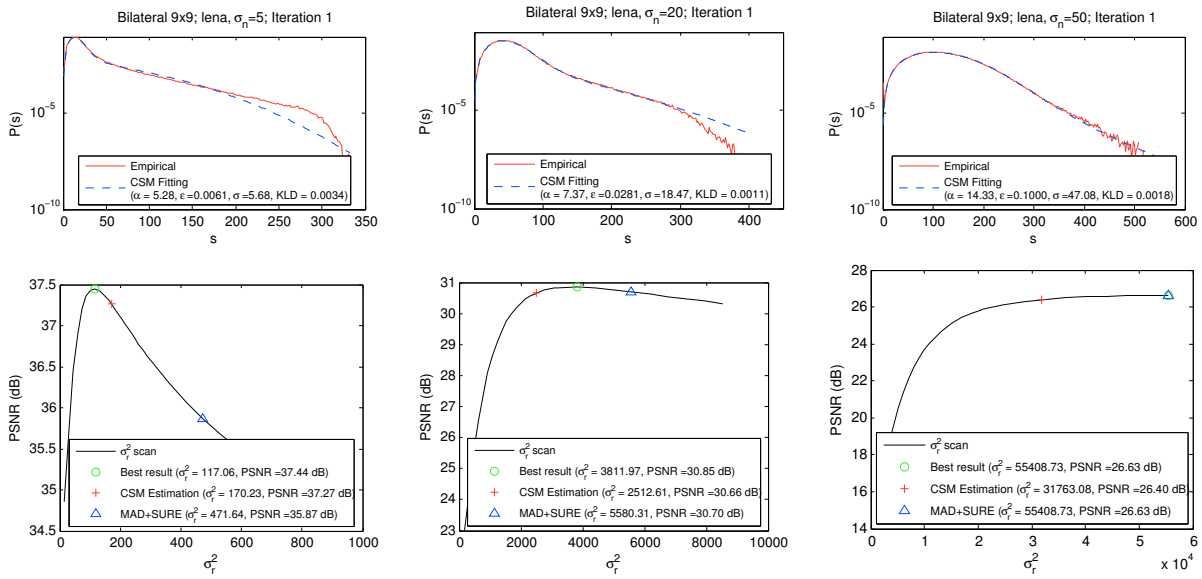


Figure 3. **BF-9×9** ($\epsilon_{bd} = 0.1$) for *Lena* with $\sigma_n = 5$, 20, and 50. The top row shows the model fitting of empirical distributions. The bottom row presents the filtered results, where the best σ_r^2 is marked by a circle, CSM estimation by a cross, and MAD+SURE by a triangle.

Table 1. Detailed result for **BF-9×9** ($\epsilon_{bd} = 0.1$). △PSNR is calculated by comparing the PSNR to the best result derived by the σ_r^2 scan, and so as $\triangle\sigma_r^2$ which is presented in form of relative percentage, i.e. $\frac{\sigma_r^2 - \sigma_{r,best}^2}{\sigma_{r,best}^2}$. The execution time is evaluated by running MATLAB on a 3.4 GHz CPU (single-thread).

Image	**σ_n=5, 1st iteration**												**σ_n=20, 1st iteration**											
	Best		CSM Fitting						MAD+SURE				Best		CSM Fitting						MAD+SURE			
	PSNR (dB)	σ_r^2	σ^2	α	KLD	ΔPSNR (dB)	$\Delta\sigma_r^2$ (rel.)	time (sec)	σ^2_{MAD}	ΔPSNR (dB)	$\Delta\sigma_r^2$ (rel.)	time (sec)	PSNR (dB)	σ_r^2	σ^2	α	KLD	ΔPSNR (dB)	$\Delta\sigma_r^2$ (rel.)	time (sec)	σ^2_{MAD}	ΔPSNR (dB)	$\Delta\sigma_r^2$ (rel.)	time (sec)
Lena	37.4	117	32.2	5.3	0.0034	**-0.2**	45%	16.1	52.8	-1.6	303%	264.0	30.9	3812	341.1	7.4	0.0011	-0.2	-34%	17.8	448.8	**-0.2**	46%	251.8
Baboon	35.0	98	70.5	3.0	0.0059	**-0.8**	115%	17.0	193.8	-5.3	949%	297.2	26.3	1725	381.2	3.3	0.0011	**-0.2**	-27%	17.9	646.2	-0.8	109%	325.9
Barbara	37.3	130	26.2	3.8	0.0011	**-0.1**	-24%	22.4	71.8	-1.9	298%	519.3	28.7	2356	326.8	4.6	0.0009	-0.4	-37%	24.8	501.5	**-0.3**	55%	548.1
Peppers	36.7	92	40.4	5.7	0.0035	**-0.7**	149%	15.0	61.9	-2.1	539%	304.4	30.7	4384	339.8	7.0	0.0020	-0.3	-45%	17.9	448.8	**-0.1**	27%	298.6
F16	38.8	134	24.7	5.5	0.0040	**0.0**	2%	13.8	44.3	-1.1	202%	329.0	31.3	3526	315.5	6.5	0.0024	-0.6	-42%	16.7	448.8	**-0.1**	46%	289.4
House	37.7	106	29.2	5.6	0.0042	**-0.2**	54%	5.9	44.3	-1.2	208%	89.7	30.8	3537	316.5	6.3	0.0043	-0.5	-43%	7.6	448.8	**-0.1**	46%	73.4
Kodim04	38.7	127	28.0	5.2	0.0113	**0.0**	15%	19.2	44.3	-1.0	147%	482.9	31.0	3280	347.2	8.1	0.0008	-0.1	-14%	25.4	423.5	**-0.1**	37%	480.9
Kodim08	37.3	127	31.1	3.9	0.0116	**0.0**	-5%	21.7	118.7	-3.1	428%	468.5	27.9	2101	332.7	4.3	0.0017	**-0.3**	-31%	25.6	586.1	-0.5	77%	481.2
Kodim13	36.8	131	41.3	3.5	0.0224	**0.0**	10%	23.4	146.5	-4.1	468%	485.8	27.4	2023	373.2	4.8	0.0009	**0.0**	-11%	25.0	615.8	-0.8	106%	478.6
Kodim19	38.5	133	26.7	5.2	0.0088	**0.0**	5%	18.3	52.8	-1.4	159%	440.0	29.9	2744	320.9	6.0	0.0018	-0.2	-30%	22.1	474.8	**-0.1**	31%	458.6
Kodim22	38.0	124	30.4	4.8	0.0158	**0.0**	17%	21.3	52.8	-1.4	174%	434.2	29.9	2860	352.9	7.5	0.0008	**0.0**	-8%	23.6	448.8	-0.1	38%	444.5
Kodim23	40.3	180	25.0	6.5	0.0065	**0.0**	-10%	20.2	36.6	-0.7	127%	437.3	32.8	4792	327.5	8.4	0.0014	-0.4	-43%	23.7	423.5	**-0.1**	24%	442.6
Average	*37.7*	*125*	*33.8*	*4.8*	*0.0082*	**-0.2**	*31%*	*17.9*	*76.7*	*-2.1*	*334%*	*379.4*	*29.8*	*3095*	*339.6*	*6.2*	*0.0016*	**-0.3**	*-30%*	*20.7*	*492.9*	*-0.3*	*53%*	*381.1*

Image	**σ_n=50, 1st iteration**												**σ_n=50, 2nd iteration**											
	Best		CSM Fitting						MAD+SURE				Best		CSM Fitting						MAD+SURE			
	PSNR (dB)	σ_r^2	σ^2	α	KLD	ΔPSNR (dB)	$\Delta\sigma_r^2$ (rel.)	time (sec)	σ^2_{MAD}	ΔPSNR (dB)	$\Delta\sigma_r^2$ (rel.)	time (sec)	PSNR (dB)	σ_r^2	σ^2	α	KLD	ΔPSNR (dB)	$\Delta\sigma_r^2$ (rel.)	time (sec)	σ^2_{MAD}	ΔPSNR (dB)	$\Delta\sigma_r^2$ (rel.)	time (sec)
Lena	26.6	55409	2216.3	14.3	0.0018	-0.2	-43%	20.1	2523.7	**0.0**	0%	295.7	27.5	439	30.0	5.3	0.0106	**-0.1**	-64%	13.1	23.9	-0.9	-97%	314.4
Baboon	21.6	17733	2459.3	10.4	0.0020	**-0.1**	44%	22.8	2836.9	-0.2	79%	330.6	21.5	69	49.8	4.4	0.0134	-0.1	213%	13.4	50.2	**0.0**	-64%	352.2
Barbara	23.7	25776	2306.5	11.8	0.0014	**0.0**	5%	27.3	2646.8	-0.1	62%	475.0	24.0	159	35.1	4.2	0.0072	**0.0**	-8%	22.3	38.1	-0.2	-89%	515.3
Peppers	26.1	46991	2181.2	11.9	0.0018	-0.3	-45%	20.5	2584.9	**0.0**	16%	306.9	27.2	698	38.6	5.3	0.0091	**-0.2**	-70%	15.4	33.0	-1.2	-97%	292.0
F16	25.8	38353	2120.4	10.4	0.0018	-0.3	-42%	20.4	2523.7	**0.0**	10%	307.5	27.0	561	43.5	5.6	0.0083	**-0.2**	-57%	15.3	41.6	-1.2	-96%	308.7
House	25.6	39298	2172.6	10.8	0.0042	-0.2	-40%	11.8	2584.9	**0.0**	19%	72.8	26.8	586	40.0	5.3	0.0072	**-0.2**	-64%	6.1	39.6	-1.2	-97%	73.7
Kodim04	26.6	54958	2198.3	14.3	0.0015	-0.2	-43%	26.5	2523.7	**0.0**	0%	457.9	27.5	410	31.7	6.0	0.0132	**-0.1**	-54%	21.4	22.4	-0.9	-96%	464.8
Kodim08	22.1	16971	2093.8	6.0	0.0018	-0.2	-26%	25.5	2772.8	**-0.1**	38%	454.7	22.9	668	98.8	4.5	0.0085	**0.0**	-33%	24.3	154.7	-0.7	-93%	477.4
Kodim13	22.2	17463	2154.5	7.2	0.0016	**0.0**	-12%	29.8	2772.8	-0.1	38%	434.1	22.7	319	78.2	4.9	0.0090	**0.0**	19%	22.7	101.6	-0.4	-88%	454.9
Kodim19	24.3	27073	2098.1	8.9	0.0015	-0.1	-31%	29.6	2584.9	**0.0**	13%	464.3	25.2	419	54.7	5.8	0.0082	**0.0**	-24%	20.9	56.8	-0.8	-93%	445.8
Kodim22	25.6	43316	2186.0	12.6	0.0015	-0.1	-37%	25.9	2584.9	**0.0**	26%	494.0	26.2	320	37.4	6.0	0.0123	**0.0**	-30%	19.8	28.8	-0.6	-94%	430.4
Kodim23	27.2	52888	2115.5	12.2	0.0017	-0.4	-51%	28.3	2523.7	**0.0**	0%	446.2	28.6	709	39.2	6.6	0.0127	**-0.2**	-64%	23.6	29.7	-1.6	-97%	459.3
Average	*24.8*	*36352*	*2191.9*	*10.9*	*0.0019*	*-0.2*	*-27%*	*24.0*	*2622.0*	**-0.1**	*25%*	*378.3*	*25.6*	*446*	*48.1*	*5.3*	*0.0100*	**-0.1**	*-20%*	*18.2*	*51.7*	*-0.8*	*-92%*	*382.4*

6.3. MNLM filter

The test results are summarized in Table 3. The **MNLM-Simple** is directly inferred from the patch-based NNM variation, and the CSM estimation can track the σ_r^2 well for different σ_n and in different iterations. The recursive **MNLM-Simple** filter can increase PSNR by up to 3.9 dB. On the other hand, the **MNLM-DCT** can provide better PSNR due

Table 2. Summary for Yaroslavsky/Bilateral filters. The numbers represent the corresponding averages for the twelve test images. $\triangle$PSNR here is calculated by comparing the PSNR to the best first-iteration result. $|\triangle\sigma_r^2|$ presents the relative percentage of absolute difference. $\bar{m}$ stands for the average number of iterations. The results of $\epsilon_{bd} = 0.1/1.0$ differ only for $\sigma_n = 40/50$.

		1st iteration								Recursive filtering			
		Best	CSM Fitting					MAD+SURE		CSM Fitting			
	σ_n	PSNR (dB)	KLD ($\times 10^{-3}$)		ΔPSNR (dB)		$\lvert\Delta\sigma_r^2\rvert$ (rel.)		$\lvert\Delta\sigma_r^2\rvert$ (rel.)	$\bar{m}$	ΔPSNR (dB)		
			ϵ_{bd} 0.1	ϵ_{bd} 1.0	ϵ_{bd} 0.1	ϵ_{bd} 1.0	ϵ_{bd} 0.1	ϵ_{bd} 1.0			ϵ_{bd} 0.1	ϵ_{bd} 1.0	ϵ_{bd} 0.1	ϵ_{bd} 1.0
YF 5x5	5	37.5	7.9		-0.2		51%	-2.0	377%	1.0	-0.2		
	10	33.4	1.5		-0.1		16%	-0.8	133%	1.1	-0.1		
	20	29.5	1.4		-0.2		25%	-0.3	57%	2.0	0.5		
	40	25.4	1.5	2.0	-0.2	-0.6	30%	43%	-0.1	31%	2.3	2.8	0.9	1.0
	50	24.1	1.5	1.7	-0.1	-0.7	31%	51%	0.0	23%	2.6	3.0	1.1	1.2
YF 9x9	5	37.5	8.8		-0.5		76%		-2.0	300%	1.0	-0.5		
	10	33.4	2.3		-0.1		32%		-0.8	133%	1.2	-0.2		
	20	29.5	1.3		-0.1		19%		-0.3	52%	2.0	0.3		
	40	25.6	1.1	1.4	-0.1	-0.2	22%	28%	-0.1	22%	2.1	2.1	0.6	0.7
	50	24.5	0.9	1.1	-0.1	-0.4	20%	34%	0.0	23%	2.1	2.1	0.6	0.8
BF 9x9	5	37.7	8.2		-0.2		38%		-2.1	334%	1.0	-0.2		
	10	33.7	1.7		-0.1		18%		-0.8	127%	1.2	-0.1		
	20	29.8	1.6		-0.3		30%		-0.3	53%	2.0	0.4		
	40	25.9	1.9	2.4	-0.3	-0.7	35%	45%	-0.1	30%	2.2	2.2	0.7	0.7
	50	24.8	1.9	2.2	-0.2	-1.1	35%	56%	-0.1	25%	2.2	3.0	0.7	0.9
BF 13x13	5	37.7	8.9		-0.3		64%		-2.0	309%	1.0	-0.3		
	10	33.6	1.9		-0.1		21%		-0.9	128%	1.2	-0.1		
	20	29.7	1.6		-0.1		22%		-0.3	48%	2.0	0.3		
	40	25.9	1.4	1.7	-0.2	-0.4	25%	32%	-0.1	25%	2.1	2.1	0.7	0.7
	50	24.7	1.2	1.5	-0.1	-0.6	26%	39%	0.0	22%	2.1	2.1	0.6	0.8

to the use of DCT-Wiener filtering, and it also terminates earlier. Though less accurate, the CSM fitting still provides good estimation of σ_r^2 for it. Note that the MAD+SURE cannot be applied here because the neighborhood is dynamically derived. For comparing the conventional NLM and MNLM, we also tested NLM in form of (2) using a σ_r^2 scan. Its best PSNR differs by less than 1% compared to **MNLM-Simple**, which indicates NLM and MNLM have similar performance.

Two state-of-the-art denoising algorithms, CPLOW [4] and CBM3D [7] ("C" stands for the versions for color images), are also tested for comparison. The MAD-derived noise variance σ_{MAD}^2 is used for them, and all other parameters are set as default in the software provided by their authors. The CBM3D performs best due to the processing in sparse representation and the good heuristic parameters. Our **MNLM-DCT** outperforms the CPLOW, though sharing a similar flow. Besides the CSM parameter optimization, one other reason is that the CPLOW processes color channels separately while we consider all channels jointly.

7. Conclusion

In this paper, we propose and study a unified framework which can both infer the neighborhood filters and estimate the range variance. With the neighborhood noise model, we show that the Yaroslavsky, bilateral, and MNLM filters can be derived by joint MAP and ML optimization. We also present an EM+ algorithm for parameter estimation via fitting the observable CSM. The experimental results on color-image denoising show the effectiveness. The noisy images can be fit well and the range variance can be tracked as accurate as the multi-pass SURE-based method. Moreover, the CSM fitting also works for filtered images, and this enables a recursive filtering scheme and improves PSNR.

The proposed framework can be used to build efficient filters for different constraints automatically, instead of heuristic tuning. It can also be expected that it will be applied to other range-weighted algorithms by formulating the corresponding likelihood functions. Therefore, we believe that it will help many computer vision problems be solved in an empirical Bayesian way, instead of an intuitive way.

Acknowledgment

This work was supported by the Ministry of Science and Technology, Taiwan, R.O.C. under Grant No. MOST 103-2218-E-007-008-MY3.

Table 3. Summary of test results for MNLM filters. $\triangle\text{PSNR}_1$ and $\triangle\text{PSNR}_2$ are relative to the first-iteration and the second-iteration best PSNR respectively. The case of $|\triangle\sigma_r^2|$ is similar. The results of CPLOW and CBM3D are given for reference.

MNLM -Simple	1st iteration				2nd iteration				Recursive filtering			**CPLOW -MAD**	**CBM3D -MAD**
	Best	CSM Fitting			Best	CSM Fitting			CSM Fitting				
σ_n	PSNR (dB)	KLD	ΔPSNR_1 (dB)	$\lvert\Delta\sigma_r^2\rvert_1$ (rel.)	PSNR (dB)	KLD	ΔPSNR_2 (dB)	$\lvert\Delta\sigma_r^2\rvert_2$ (rel.)	$\bar{m}$	ΔPSNR_1 (dB)	PSNR (dB)	PSNR (dB)	PSNR (dB)
5	36.5	0.0047	-0.2	38%					1.4	-0.3	36.2		
10	32.5	0.0008	-0.2	29%					1.3	-0.2	32.3		
20	28.5	0.0005	-0.1	20%	29.8	0.0048	-0.03	22%	2.3	1.3	29.8		
40	23.9	0.0002	0.0	39%	26.6	0.0015	-0.04	29%	3.0	3.2	27.1		
50	22.2	0.0002	0.0	51%	25.5	0.0008	-0.03	34%	3.0	3.9	26.1		
MNLM -DCT	1st iteration				2nd iteration				Recursive filtering			**CPLOW -MAD**	**CBM3D -MAD**
	Best	CSM Fitting			Best	CSM Fitting			CSM Fitting				
σ_n	PSNR (dB)	KLD	ΔPSNR_1 (dB)	$\lvert\Delta\sigma_r^2\rvert_1$ (rel.)	PSNR (dB)	KLD	ΔPSNR_2 (dB)	$\lvert\Delta\sigma_r^2\rvert_2$ (rel.)	$\bar{m}$	ΔPSNR_1 (dB)	PSNR (dB)	PSNR (dB)	PSNR (dB)
5	37.6	0.0047	-0.3	330%					1.0	-0.3	37.3	33.6	37.1
10	34.8	0.0008	-0.3	131%					1.0	-0.3	34.4	31.8	34.7
20	31.6	0.0012	-0.2	81%					1.0	-0.2	31.3	29.6	32.0
40	28.1	0.0003	-0.1	47%	28.6	0.0107	-0.01	62%	2.0	0.5	28.6	27.2	29.2
50	27.0	0.0004	-0.1	57%	27.7	0.0068	-0.02	60%	2.0	0.7	27.7	26.4	28.3

References

[1] D. Barash and D. Comaniciu. A common framework for nonlinear diffusion, adaptive smoothing, bilateral filtering and mean shift. *Image and Vision Computing*, 22(1):73–81, Jan 2004. 1

[2] A. Buades, B. Coll, and J. M. Morel. A review of image denoising algorithms, with a new one. *SIAM Journal on Multiscale Modeling and Simulation*, 4(2):490–530, 2005. 1, 2

[3] A. Buades, B. Coll, and J.-M. Morel. The staircasing effect in neighborhood filters and its solution. *IEEE Transactions on Image Processing*, 15(6):1499–1505, Jun 2006. 1

[4] P. Chatterjee and P. Milanfar. Patch-based near-optimal image denoising. *IEEE Transactions on Image Processing*, 21(4):1635–1649, Apr 2012. 2, 6, 8

[5] Y. Chen and K. J. R. Liu. Image denoising games. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(10):1704–1716, oct 2013. 2

[6] P. Choudhury and J. Tumblin. The trilateral filter for high contrast images and meshes. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, New York, NY, USA, 2005. ACM. 1

[7] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, Aug 2007. 2, 8

[8] M. Elad. On the origin of the bilateral filter and ways to improve it. *IEEE Transactions on Image Processing*, 11(10):1141–1151, Oct 2002. 1

[9] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with application to image denoising. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2862–2869, 2014. 2

[10] M. Jamshidian and R. I. Jennrich. Acceleration of the EM algorithm by using quasi-newton methods. *Journal of the Royal Statistical Society. Series B (Methodological)*, 59(3):569–587, 1997. 4

[11] C. Kervrann and J. Boulanger. Optimal spatial adaptation for patch-based image denoising. *IEEE Transactions on Image Processing*, 15(10):2866–2878, Oct 2006. 1

[12] H. Kishan and C. S. Seelamantula. SURE-fast bilateral filters. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pages 1129–1132, 2012. 2

[13] C. Liu, R. Szeliski, S. B. Kang, C. L. Zitnick, and W. T. Freeman. Automatic estimation and removal of noise from a single image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):299–314, Feb 2008. 2

[14] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *IEEE International Conference on Computer Vision*, pages 2272–2279, 2009. 2

[15] D. P. McMillen and J. F. McDonald. Locally weighted maximum likelihood estimation: Monte carlo evidence and an application. In *Advances in Spatial Econometrics*, pages 225–239. Springer Berlin Heidelberg, 2004. 3

[16] P. Milanfar. A tour of modern image filtering: New insights and methods, both practical and theoretical. *IEEE Signal Processing Magazine*, 30(1):106–128, Jan 2013. 1

[17] S. Paris, P. Kornprobst, J. Tumblin, and F. Durand. Bilateral filtering: Theory and applications. In *Foundations and Trends in Computer Graphics and Vision*, volume 4, pages 1–73, 2008. 1

[18] H. Peng and R. Rao. Bilateral kernel parameter optimization by risk minimization. In *Proceedings of International Conference on Image Processing*, pages 3293–3296, 2010. 2

[19] H. Peng, R. Rao, and S. A. Dianat. Multispectral image denoising with optimized vector bilateral filter. *IEEE Transactions on Image Processing*, 23(1):264–273, Jan 2014. 2

[20] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama. Digital photography with flash and no-flash image pairs. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 664–672, New York, NY, USA, 2004. ACM. 1

[21] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli. Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE Transactions on Image Processing*, 12(11):1338–1351, Nov 2003. 2

[22] C. M. Stein. Estimation of the mean of a multivariate normal distribution. *The Annals of Statistics*, 9(6):1135–1151, 1981. 1

[23] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of International Conference on Computer Vision*, pages 839–846, 1998. 1, 2

[24] D. V. D. Ville and M. Kocher. SURE-based non-local means. *IEEE Signal Processing Letters*, 16(11):973–976, Nov 2009. 2

[25] D. V. D. Ville and M. Kocher. Nonlocal means with dimensionality reduction and SURE-based parameter selection. *IEEE Transactions on Image Processing*, 20(9):2683–2690, sep 2011. 2

[26] M. J. Wainwright and E. P. Simoncelli. Scale mixtures of Gaussian and the statistics of natural images. *Advaned Neural Information Processing Systems*, 12:855–861, May 2000. 2

[27] L. Yaroslavsky, K. Egiazarian, and J. Astola. Transform domain image restoration methods: review, comparison, and interpretation. In *Nonlinear Image Processing and Pattern Analysis XII*, volume 4304 of *Proc. SPIE*, pages 155–169, 2001. 2

[28] L. P. Yaroslavsky. *Digital Picture Processing - An Introduction*. Springer Verlag, 1985. 1, 2