Optimal Combination of Image Denoisers

Joon Hee Choi, *Member, IEEE*, Omar A. Elgendy, *Student Member, IEEE*, and Stanley H. Chan, *Senior Member, IEEE*

***Abstract*— Given a set of image denoisers, each having a different denoising capability, is there a provably optimal way of combining these denoisers to produce an overall better result? An answer to this question is fundamental to designing an ensemble of weak estimators for complex scenes. In this paper, we present an optimal combination scheme by leveraging the deep neural networks and the convex optimization. The proposed framework, called the Consensus Neural Network (CsNet), introduces three new concepts in image denoising: 1) a provably optimal procedure to combine the denoised outputs via convex optimization; 2) a deep neural network to estimate the mean squared error (MSE) of denoised images without needing the ground truths; and 3) an image boosting procedure using a deep neural network to improve the contrast and to recover the lost details of the combined images. Experimental results show that CsNet can consistently improve the denoising performance for both deterministic and neural network denoisers.**

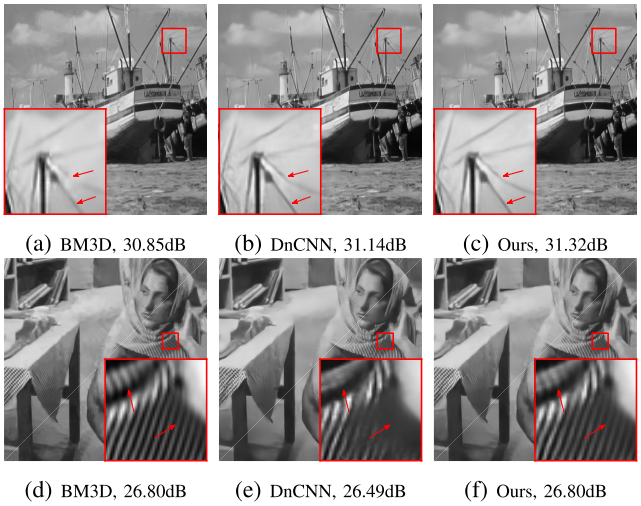
***Index Terms*— Image denoising, optimal combination, convex optimization, deep learning, convolutional neural networks.**

I. Introduction

WHILE image denoising algorithms over the past decade have produced very promising results, it is also safe to say that no single method is uniformly better than others. In fact, any image denoiser, either deterministic [1]–[8] or learning-based [9]–[22], has an implicit prior model that determines its denoising characteristics. Since a particular prior model encapsulates the statistics of a limited set of imaging conditions, the corresponding denoiser is only an expert for the type of images it is designed to handle. We refer to this gap between the denoising model and the denoising task as a model mismatch.

Model mismatch is common in practice. In this paper, we are particularly interested in the following three examples:

- Denoiser Characteristic: Every denoiser has a different characteristic. For example, BM3D [2] assumes patch reoccurrence, and thus it works well for images with repeated patterns. Neural network denoisers are trained on generic images, and thus they work well for those images. Figure 1 shows an example of BM3D [2] and a neural network denoiser DnCNN [19]. The `Boat512` image is corrupted by i.i.d. Gaussian noise of noise level $\sigma = 20$. In this example, DnCNN (trained at $\sigma = 20$) gives a PSNR of 31.14dB which is approximately 0.3dB higher than BM3D. The other image `Barbara512` is corrupted by a noise of level $\sigma = 40$. In this case, BM3D actually performs better than DnCNN (trained at $\sigma = 40$), yielding 26.80dB over the 26.49dB. If we look at the image content, we can see that `Barbara512` has a repeated pattern on the cloth which is more favorable to BM3D. This shows the influence of the implicit modelings of a denoiser to the performance.
- Noise Level: For neural network image denoisers, the performance is strongly affected by the noise level under which the denoiser is trained. For example, if a denoiser is trained for i.i.d. Gaussian noise of standard deviation σ, it only works well for this particular σ. As soon as the noise level deviates, the performance will degrade. The



(a) BM3D, 30.85dB (b) DnCNN, 31.14dB (c) Ours, 31.32dB

(d) BM3D, 26.80dB (e) DnCNN, 26.49dB (f) Ours, 26.80dB

Fig. 1. Comparison of BM3D [2], DnCNN [19] and the proposed CsNet. The `boat` image is corrupted by noise of $\sigma = 20$, whereas `Barbara` is corrupted by noise of $\sigma = 40$. The denoising strength of the denoisers are adjusted to match the actual noise level. The results show that different denoisers are better for different types of images, e.g., BM3D is better for repeated pattern whereas DnCNN is better for generic content. The combination scheme proposed in this paper is able to leverage the better among the two. (a) BM3D, 30.85dB. (b) DnCNN, 31.14dB. (c) Ours, 31.32dB. (d) BM3D, 26.80dB. (e) DnCNN, 26.49dB. (f) Ours, 26.80dB.

Manuscript received May 24, 2018; revised September 1, 2018 and January 27, 2019; accepted February 26, 2019. Date of publication March 8, 2019; date of current version June 20, 2019. This work was supported in part by the U.S. National Science Foundation under Grant CCF-1718007 and Grant CCF-1763896. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Weisheng Dong. *(Corresponding author: Stanley H. Chan.)*

J. H. Choi was with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA. He is now with Qualcomm Inc., San Diego, CA 92121 USA (e-mail: choi240@purdue.edu).

O. A. Elgendy is with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: oelgendy@purdue.edu).

S. H. Chan is with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA, and also with the Department of Statistics, Purdue University, West Lafayette, IN 47907 USA (e-mail: stanchan@purdue.edu).

Digital Object Identifier 10.1109/TIP.2019.2903321

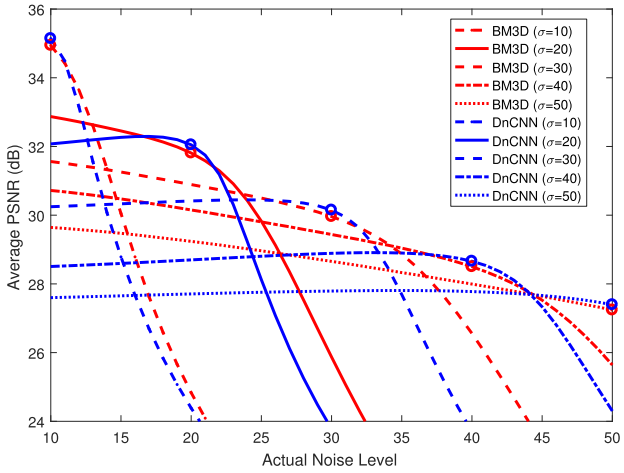


Fig. 2. Illustration of noise-level mismatch. We compare BM3Ds and DnCNNs at noise levels $\widehat{\sigma} \in \{10, 20, 30, 40, 50\}$ in terms of true noise levels $[10, 50]$ on 10 Kodak images.

same argument holds for deterministic denoisers such as BM3D, as its denoising strength must match the actual noise level. Figure 2 illustrates the behavior of DnCNN and BM3D as the denoising strength $\widehat{\sigma}$ deviates from the actual level σ. In this experiment, we use five denoising strengths $\widehat{\sigma} = \{10, 20, 30, 40, 50\}$ and a continuous range of $\sigma \in [10, 50]$. As shown in the plot, BM3D has a slightly more robust performance, in the sense that a chosen denoising strength $\widehat{\sigma}$ can work for a reasonable wide range of actual noise levels σ. In contrast, DnCNN has a narrow performance regime for a fixed $\widehat{\sigma}$.

- Image Class: A denoiser trained for a particular class of images (e.g., building) may not work for other classes (e.g., face). When this type of class-aware issue appears, the typical solution is by means of scene classification [18]. However, scene classification itself is an open problem and there is no consensus of the best approach. Therefore, it would be more convenient if the denoiser can automatically pick a class that gives the best performance without seeking classification algorithms.

The examples above bring out a question that if we have a set of denoisers, each having a different characteristic, how do we combine them to produce a better result? Answering this question is fundamental to designing ensembles of expert image restoration methods for complex scenes. The goal of this paper is to present a framework called the Consensus Neural Network (CsNet) which seeks consensus by using neural networks and convex optimization.

A. Related Work

Combining estimators is a long-standing statistical problem. In as early as 1959, Graybill and Deal [23] started to consider linearly combining two unbiased scalar estimators to yield a new estimator that remains unbiased and has lower variance. More properties of the such combination scheme were discussed by Samuel-Cahn [24]. Rubin and Weisberg [25] extended the idea by estimating weights from the samples.

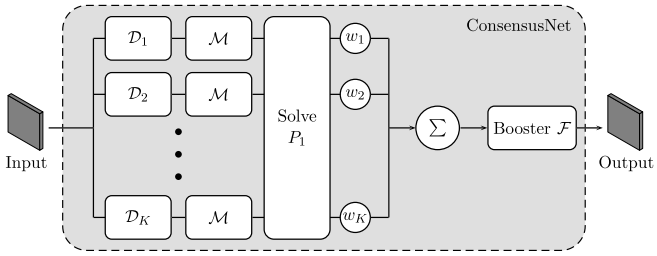


Fig. 3. Structure of the proposed CsNet: Given a set of K initial denoisers $\mathcal{D}_1, \ldots, \mathcal{D}_K$, CsNet uses an MSE estimator ($\mathcal{M}$) to estimate the MSE of each initial denoiser. After the MSEs are estimated, we solve a convex optimization problem (P_1) to determine the optimal weight $w_1, \ldots, w_K$. The combined estimate is then boosted using a booster neural network to improve contrast and details.

However, the estimators are still scalars and are assumed to be independent. Correlated scalar estimators are later studied by Keller and Olkin [26]. For vector estimators (which is the case for image denoisers), Odell *et al.* [27] presented a very comprehensive study. However, their result is limited to two vector estimators. The general case of multiple estimators is studied by Lavancier and Rochet [28], who proposed an optimization approach to estimate the weights.

Specific to image denoising, methods seeking linear combination of denoisers are scattered in the literature. The most popular approach is perhaps the linear expansion of thresholds by Blu and colleagues [29], using the Stein's unbiased risk estimator (SURE). Chaudhury and Rithwik [30] presented an improved bilateral filter using the SURE estimator. For learning based methods, the loss-specific training approach by Jancsary *et al.* [31] presented a regression tree field model to optimize the denoising performance over different metrics. There is also an end-to-end neural network solution for selecting denoisers by Agostinelli *et al.* [32], where the authors proposed to learn the weights using an auto-encoder.

The noise-level mismatch is discussed more often in the neural network literature. Conventional approach is to either truncate the noise level to the nearest trained level [33] or to train the network with a large number of examples covering all noise levels [19]. A more recent approach is to feed a noise map to the network and train the network to recognize the noise level [21]. However, this approach requires a redesign of the network structure. In contrast, CsNet uses the same structure for all initial denoisers.

B. Contributions

An overview of the proposed CsNet framework is shown in Figure 3. We summarize the three key contributions of this paper in the followings:

- Optimal Combination. We present an optimal combination framework via convex optimization. By minimizing a quadratic function over a unit simplex, we prove that the resulting combination is optimal in the MSE sense. We provide geometric interpretation of the solution, and a fast algorithm to determine the optimal point.
- MSE Estimator. We present a novel deep neural network to estimate the mean square error (MSE) in the absence

of the ground truth. Existing deep neural network-based image quality assessment methods are designed to predict perceptual quality and not MSE. To the best of our knowledge, our deep learning based MSE estimator is the first of this kind in the literature.

- Denoising Booster. We present a new deep neural network to boost the combined estimates. Unlike the existing boosters which are iterative, we cascade multiple simple neural networks to achieve a one-shot booster.

To help readers understand the design process, we proceed the paper by first discussing the optimal combination and its associated theoretical properties in Section II. Section III discusses the neural network estimator for estimating the MSE. We emphasize that the neural network presented here is just one of the many possible ways of estimating the MSE. Readers preferring non-training based approaches can use estimators such as SURE, although we will provide examples where SURE does not work. Section IV discusses the booster, and its cascade structure. Experiments are discussed in Section V.

C. Notation

Throughout this paper, we use lower case bold letters to denote vectors, e.g., $\boldsymbol{x} \in \mathbb{R}^N$, and upper case bold letters to denote matrices, e.g., $\boldsymbol{A} \in \mathbb{R}^{K\times K}$. An all-one vector is denoted as $\mathbf{1}$. Standard basis vectors are denoted as $\boldsymbol{e}_i$, i.e., $\boldsymbol{e}_i = [0,\ldots,1,\ldots,0]^T$. For any vector $\boldsymbol{x}$, $\|\boldsymbol{x}\|_2$ means the ℓ_2-Euclidean norm, and for any matrix $\boldsymbol{A}$, $\|\boldsymbol{A}\|_2 = \max_{\|\boldsymbol{x}\|_2=1} \|\boldsymbol{A}\boldsymbol{x}\|_2$ denotes the matrix operator norm. To specify that a vector $\boldsymbol{x}$ is non-negative for all its elements, we write $\boldsymbol{x} \succeq 0$. For matrices, $\boldsymbol{A} \succeq 0$ means that $\boldsymbol{A}$ is positive semi-definite. Images in this paper are normalized so that every pixel is in $[0, 1]$. Noise level of an i.i.d. Gaussian noise is specified by its standard deviation σ. For notational simplicity, we write σ in the scale of $[0, 255]$, e.g., "$\sigma = 20$" means $\sigma = 20/255$. Finally, an image denoiser $\mathcal{D}$ is a mapping $\mathcal{D} : [0,1]^N \to [0,1]^N$. We assume $\mathcal{D}$ is bounded and is asymptotically invariant [34].

II. Optimal Combination of Estimators

A. Problem Formulation

Consider a linear forward model where a clean image $\boldsymbol{z} \in \mathbb{R}^N$ is corrupted by additive i.i.d. Gaussian noise $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2\boldsymbol{I})$ so that the observed image is $\boldsymbol{y} = \boldsymbol{z} + \boldsymbol{\eta}$. We apply a set of K image denoisers $\mathcal{D}_1, \ldots, \mathcal{D}_K$ to yield K initial estimates $\widehat{\boldsymbol{z}}_k = \mathcal{D}_k(\boldsymbol{y})$ for $k = 1, \ldots, K$. For convenience, we concatenate these initial estimates by constructing a matrix $\widehat{\boldsymbol{Z}} = [\widehat{\boldsymbol{z}}_1, \ldots, \widehat{\boldsymbol{z}}_K] \in \mathbb{R}^{N\times K}$.

In this paper, we are interested in the *linear combination* of estimators. That is, for a given $\widehat{\boldsymbol{Z}}$, we construct the linearly combined estimate as

$$\widehat{\boldsymbol{z}} = \sum_{k=1}^{K} w_k \widehat{\boldsymbol{z}}_k = \widehat{\boldsymbol{Z}}\boldsymbol{w}, \tag{1}$$

where $\boldsymbol{w} \overset{\text{def}}{=} [w_1, \ldots, w_K]^T \in \mathbb{R}^K$ is the vector of combination weights. The goal of our work is to formulate an optimization problem to determine the optimal weights.

For analytic tractability, we use mean squared error (MSE) to measure the optimality, although it is known that alternative visual quality metrics correlate better to human visual systems [35]. Denoting $\boldsymbol{z} \in \mathbb{R}^N$ as the ground truth, we define the MSE between the combined estimate $\widehat{\boldsymbol{z}}$ and the ground truth $\boldsymbol{z}$ as

$$\text{MSE}(\widehat{\boldsymbol{z}}, \boldsymbol{z}) \overset{\text{def}}{=} \mathbb{E}\left[\|\widehat{\boldsymbol{z}} - \boldsymbol{z}\|^2\right] = \mathbb{E}\left[\left\|\widehat{\boldsymbol{Z}}\boldsymbol{w} - \boldsymbol{z}\right\|^2\right]. \tag{2}$$

The optimal combination problem can be posed as minimizing the MSE by seeking the weight vector $\boldsymbol{w} \in \mathbb{R}^K$:

$$\begin{aligned} &\underset{\boldsymbol{w}}{\text{minimize}} \;\; \mathbb{E}\left[\|\widehat{\boldsymbol{Z}}\boldsymbol{w} - \boldsymbol{z}\|^2\right] \\ &\text{subject to} \;\; \boldsymbol{w}^T\mathbf{1} = 1, \quad \text{and } \boldsymbol{w} \succeq 0. \end{aligned} \tag{3}$$

Here, the constraint $\boldsymbol{w}^T\mathbf{1} = 1$ ensures that the sum of the weights is 1, and the constraint $\boldsymbol{w} \succeq 0$ ensures that the combined estimate remains in $[0, 1]^N$.

Let us simplify (3). First, we define $\boldsymbol{Z} = [\boldsymbol{z}, \ldots, \boldsymbol{z}] \in \mathbb{R}^{N\times K}$, i.e., a matrix with the ground truth $\boldsymbol{z}$ in each column. Since $\boldsymbol{w}^T\mathbf{1} = 1$, we can show that

$$\begin{aligned} \mathbb{E}\left[\left\|\widehat{\boldsymbol{Z}}\boldsymbol{w} - \boldsymbol{z}\right\|^2\right] &= \mathbb{E}\left[\left\|\widehat{\boldsymbol{Z}}\boldsymbol{w} - \boldsymbol{Z}\boldsymbol{w}\right\|^2\right] \\ &= \mathbb{E}\left[\boldsymbol{w}^T(\widehat{\boldsymbol{Z}} - \boldsymbol{Z})^T(\widehat{\boldsymbol{Z}} - \boldsymbol{Z})\boldsymbol{w}\right] \\ &= \boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}, \end{aligned}$$

where $\boldsymbol{\Sigma}$ is defined as

$$\boldsymbol{\Sigma} \overset{\text{def}}{=} \mathbb{E}\left[(\widehat{\boldsymbol{Z}} - \boldsymbol{Z})^T(\widehat{\boldsymbol{Z}} - \boldsymbol{Z})\right].$$

We call $\boldsymbol{\Sigma}$ the *covariance matrix*.[1] Using this result, we can rewrite (3) into an equivalent form as

$$\begin{aligned} &\underset{\boldsymbol{w}}{\text{minimize}} \;\; \boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w} \\ &\text{subject to} \;\; \boldsymbol{w}^T\mathbf{1} = 1, \quad \text{and } \boldsymbol{w} \succeq 0, \end{aligned} \tag{P_1}$$

which is a convex problem because $\boldsymbol{\Sigma}$ is positive semi-definite and the feasible set is convex.

Before we discuss how to solve (P_1), we should first discuss how to obtain $\boldsymbol{\Sigma}$. The (i, i)-th entry of $\boldsymbol{\Sigma}$ is

$$\Sigma_{ii} = \mathbb{E}\left[\left\|\widehat{\boldsymbol{z}}_i - \boldsymbol{z}\right\|^2\right] \overset{\text{def}}{=} \text{MSE}_i,$$

which is the MSE of the i-th estimate. The (i, j)-th entry of $\boldsymbol{\Sigma}$ is the correlation between $\widehat{\boldsymbol{z}}_i$ and $\widehat{\boldsymbol{z}}_j$:

$$\Sigma_{ij} = \mathbb{E}\left[(\widehat{\boldsymbol{z}}_i - \boldsymbol{z})^T(\widehat{\boldsymbol{z}}_j - \boldsymbol{z})\right].$$

To express Σ_{ij} in terms of MSE_i and MSE_j, we notice that

$$\begin{aligned} \mathbb{E}\left[\left\|\widehat{\boldsymbol{z}}_i - \widehat{\boldsymbol{z}}_j\right\|^2\right] &= \mathbb{E}\left[\left\|\widehat{\boldsymbol{z}}_i - \boldsymbol{z} + \boldsymbol{z} - \widehat{\boldsymbol{z}}_j\right\|^2\right] \\ &= \mathbb{E}\left\|\widehat{\boldsymbol{z}}_i - \boldsymbol{z}\right\|^2 + \mathbb{E}\left\|\widehat{\boldsymbol{z}}_j - \boldsymbol{z}\right\|^2 \\ &\quad -2\mathbb{E}\left[(\widehat{\boldsymbol{z}}_i - \boldsymbol{z})^T(\widehat{\boldsymbol{z}}_j - \boldsymbol{z})\right] \\ &= \text{MSE}_i + \text{MSE}_j - 2\Sigma_{ij}. \end{aligned}$$

[1] Straightly speaking, $\boldsymbol{\Sigma} \overset{\text{def}}{=} \mathbb{E}\left[(\widehat{\boldsymbol{Z}} - \boldsymbol{Z})^T(\widehat{\boldsymbol{Z}} - \boldsymbol{Z})\right]$ is not the conventional covariance matrix because denoisers are not necessarily unbiased, i.e., $\mathbb{E}[\widehat{\boldsymbol{Z}}] \neq \boldsymbol{Z}$.

Rearranging the terms we can write Σ_{ij} as

$$\Sigma_{ij} = \frac{\text{MSE}_i + \text{MSE}_j - \mathbb{E}\left[\left\|\widehat{\boldsymbol{z}}_i - \widehat{\boldsymbol{z}}_j\right\|^2\right]}{2}. \tag{4}$$

Therefore, when we do not have true MSE_i and MSE_j but estimates $\widetilde{\text{MSE}}_i$ and $\widetilde{\text{MSE}}_j$, (4) provides a convenient way to construct Σ_{ij} because $\mathbb{E}\left[\left\|\widehat{\boldsymbol{z}}_i - \widehat{\boldsymbol{z}}_j\right\|^2\right]$ does not require the ground truth.

B. Solving (P_1)

The optimization problem in (P_1) is a quadratic minimization over a unit simplex. The problem does not have a closed form solution because the KKT conditions involve a complementary slackness term due to the non-negativity constraint. Iterative algorithms are available though, e.g., using general purpose semi-definite programming such as CVX [36], [37], or using projected gradients [38], [39]. However, since (P_1) has a simple structure, efficient algorithms can be derived.

Our algorithm is an accelerated gradient method following the work of Jaggi [40]. We briefly describe the algorithm for completeness. Let

$$f(\boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{\Sigma} \boldsymbol{w} \tag{5}$$

be the objective function, and

$$\Omega \stackrel{\text{def}}{=} \{\boldsymbol{w} \mid \boldsymbol{w}^T \mathbf{1} = 1, \text{ and } \boldsymbol{w} \succeq 0\} \tag{6}$$

be the feasible set. The first order linear approximation at the t-th iterate is

$$f(\boldsymbol{u}) = f(\boldsymbol{w}^{(t)}) + \nabla f(\boldsymbol{w}^{(t)})^T (\boldsymbol{u} - \boldsymbol{w}^{(t)}), \quad \forall \boldsymbol{u} \in \Omega.$$

Thus, for any $\boldsymbol{u} \in \Omega$, $\boldsymbol{u} - \boldsymbol{w}^{(t)}$ is a feasible search direction. One choice of $\boldsymbol{u}$ is to make $\nabla f(\boldsymbol{w}^{(t)})^T \boldsymbol{u}$ minimized so that $f(\boldsymbol{u})$ has a lower cost. This leads to

$$\underset{\boldsymbol{u} \in \Omega}{\text{minimize}} \quad \nabla f(\boldsymbol{w}^{(t)})^T \boldsymbol{u}, \tag{7}$$

which has a linear objective function. Once $\boldsymbol{u}$ is determined, we construct a standard accelerated gradient step:

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \alpha(\boldsymbol{u} - \boldsymbol{w}^{(t)}), \tag{8}$$

where $\alpha = \frac{2}{t+2}$ is the step size.

It remains to find the solution of the subproblem (7). However, the subproblem (7) is a linear programming over the unit simplex. Therefore, the solution has to lie on one of the vertices. We derive a closed-form solution in Proposition 1. The pseudo-code is shown in Algorithm 1.

Proposition 1: The solution to (7) *is* $\boldsymbol{u} = \boldsymbol{e}_{i^*}$, *where* $i^* = \text{argmin}_i (\nabla f(\boldsymbol{w}^{(t)}))_i$.

Proof: Let $\boldsymbol{g} = \nabla f(\boldsymbol{w}^{(t)})$. Then it follows that

$$\boldsymbol{g}^T \boldsymbol{u} = \textstyle\sum_{i=1}^K g_i u_i \geq g_{\min} \sum_{i=1}^K u_i = g_{\min},$$

where $g_{\min} = \min_i g_i$, and $\sum_{i=1}^K u_i = 1$ because $\boldsymbol{u} \in \Omega$. The lower bound can be attained when $\boldsymbol{u} = \boldsymbol{e}_{i^*}$, where $i^* = \text{argmin}_i \ g_i$. □

Example 1: As an illustration of Algorithm 1, we compare its performance with an ADMM algorithm by Condat [38].

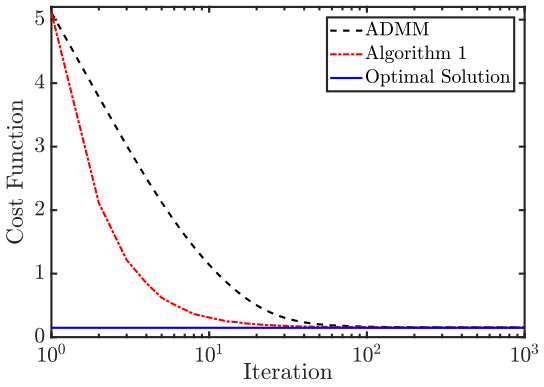


Fig. 4. Comparison of Algorithm 1 and the ADMM algorithm by [38], using the optimal solution obtained by CVX [36].

Algorithm 1 Algorithm to Solve (P_1)

1: Initialize $\boldsymbol{w}^0 = \boldsymbol{e}_1$.
2: **for** $t = 0, 1, \ldots, T_{\max}$ **do**
3: Let $i^* = \underset{i}{\text{argmin}} \ (\boldsymbol{\Sigma}\boldsymbol{w}^{(t)})_i$
4: Update $\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \left(\frac{2}{t+2}\right)(\boldsymbol{e}_{i^*} - \boldsymbol{w}^{(t)})$.
5: **end for**

The reference method is CVX [36]. We repeat the experiment 1000 *times using different random matrices* Σ, *and take the average. As shown in Figure 4, Algorithm 1 converges significantly faster than [38]. In terms of runtime, Algorithm 1 takes about* 4.4 *msec, [38] takes* 13 *msec, and CVX takes* 223.1 *msec.*

C. Geometric Interpretation of (P_1)

1) Uniqueness: The uniqueness of the solution of (P_1) is determined by the positive definiteness of $\boldsymbol{\Sigma}$. If $\boldsymbol{\Sigma}$ is positive definite, then (P_1) is strictly convex, and hence the optimal weight is unique. If $\boldsymbol{\Sigma}$ is only positive semi-definite, then there are infinitely many optimal weights. The following proposition explains this phenomenon.

Proposition 2: Suppose that $\boldsymbol{\Sigma}$ *is positive semi-definite. Let* $\boldsymbol{w}_1^*$ *and* $\boldsymbol{w}_2^*$ *be two solutions of* (P_1). *Then, for any* $0 \leq t \leq 1$, *the vector* $\boldsymbol{w}^* \stackrel{def}{=} t\boldsymbol{w}_1^* + (1-t)\boldsymbol{w}_2^*$ *is also a solution of* (P_1).

Proof: Let $f(\boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{\Sigma} \boldsymbol{w}$. Since both $\boldsymbol{w}_1^*$ and $\boldsymbol{w}_2^*$ are solutions to (P_1), we have $f(\boldsymbol{w}_1^*) = f(\boldsymbol{w}_2^*)$. Also, by linearity, we have that $\mathbf{1}^T \boldsymbol{w}^* = 1$ and $\boldsymbol{w}^* \succeq 0$. Since f is convex, we can show that

$$\begin{aligned} f(\boldsymbol{w}^*) &= f(t\boldsymbol{w}_1^* + (1-t)\boldsymbol{w}_2^*) \\ &\leq t f(\boldsymbol{w}_1^*) + (1-t) f(\boldsymbol{w}_2^*) = f(\boldsymbol{w}_1^*). \end{aligned}$$

But since $\boldsymbol{w}_1^*$ is an optimal solution, it is impossible for $f(\boldsymbol{w}^*) < f(\boldsymbol{w}_1^*)$. So the only possibility is $f(\boldsymbol{w}^*) = f(\boldsymbol{w}_1^*)$. This implies that $\boldsymbol{w}^*$ is also a solution. □

The implication of Proposition 2 is that if two initial estimates $\widehat{\boldsymbol{z}}_i$ and $\widehat{\boldsymbol{z}}_j$ are identical (or scalar multiple of one and the other), then $\boldsymbol{\Sigma}$ will have dependent columns (hence positive semi-definite). When this happens, there will be infinitely

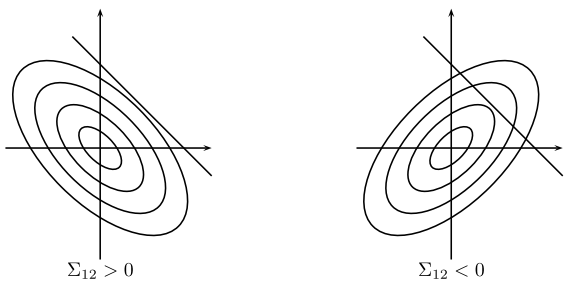


Fig. 5. Geometry of the optimal weight minimization problem.

many ways of combining the two initial estimates. However, in practice this is not an issue because even if the pair (w_i^*, w_j^*) is not unique, the combined estimate $w_i^*\widehat{z}_i + w_j^*\widehat{z}_j$ remains unique when $\widehat{z}_i = \widehat{z}_j$.

2) Geometry: The geometry of (P_1) can be interpreted in low dimensions, e.g., Figure 5. In this figure, we consider a 2D case so that $\boldsymbol{\Sigma}$ is a 2×2 matrix. We can show that the ellipse always has its minor axis pointing to the northeast direction if the two initial estimates are positively correlated.

Proposition 3: Consider a two-dimensional $\boldsymbol{\Sigma}$. *If* $\Sigma_{12} > 0$, *then* $\boldsymbol{\Sigma}$ *always has its minor axis pointing to the northeast direction and major axis to the northwest direction.*

Proof: Consider the eigen-decomposition of $\boldsymbol{\Sigma} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^T$. For a 2×2 matrix, classical results in matrix analysis [41] shows that the eigen-value and eigen-vectors are

$$s_1 = \frac{1}{2}\left(\Sigma_{11} + \Sigma_{22} + \lambda\right), \quad s_2 = \frac{1}{2}\left(\Sigma_{11} + \Sigma_{22} - \lambda\right),$$

and

$$\boldsymbol{u}_1 = \begin{bmatrix} \frac{\Sigma_{11} - \Sigma_{22} + \lambda}{2\Sigma_{12}} \\ 1 \end{bmatrix}, \quad \boldsymbol{u}_2 = \begin{bmatrix} \frac{\Sigma_{11} - \Sigma_{22} - \lambda}{2\Sigma_{12}} \\ 1 \end{bmatrix}$$

where $\lambda = \sqrt{4\Sigma_{12}^2 + (\Sigma_{11} - \Sigma_{22})^2}$.

Note that $\lambda \geq |\Sigma_{11} - \Sigma_{22}|$ because $\Sigma_{12}^2 \geq 0$. Therefore, $s_1 \geq s_2$ and so $\boldsymbol{u}_1$ is the minor axis and $\boldsymbol{u}_2$ is the major axis. The numerator of the first entry of $\boldsymbol{u}_1$ is

$$\begin{aligned} \Sigma_{11} - \Sigma_{22} + \lambda &\geq \Sigma_{11} - \Sigma_{22} + |\Sigma_{11} - \Sigma_{22}| \\ &= \begin{cases} 2|\Sigma_{11} - \Sigma_{22}| \geq 0, & \text{if } \Sigma_{11} \geq \Sigma_{22}, \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

As a result, the numerator of the first entry of $\boldsymbol{u}_1$ is always non-negative, implying that the sign of the denominator determines the sign of the entry. Therefore, if $\Sigma_{12} > 0$, then $\boldsymbol{u}_1$ will be pointing to the northeast direction. By orthogonality of the eigen-vectors, $\boldsymbol{u}_2$ points to the northwest direction. □

Proposition 3 provides some insights about the solution. If $\Sigma_{12} > 0$ (which is usually the case), the major axis must point to northwest. Therefore, the solution is more likely to be at one of the two vertices. In other words, the optimal solution tends to be *sparse*. Such sparsity should come with no surprise, because the linear constraint $\boldsymbol{w}^T\mathbf{1} = 1$ is equivalent to $\|\boldsymbol{w}\|_1 = 1$ if $\boldsymbol{w} \succeq 0$. This also explains why the non-negativity constraint in our problem is essential.

Remark 1: In practice, if we only have an estimated covariance matrix $\widetilde{\boldsymbol{\Sigma}}$, *there is no guarantee that* $\widetilde{\boldsymbol{\Sigma}}$ *is positive semi-definite. (Symmetry can be preserved by constructing the off-diagonals using* (4)*.) When* $\widetilde{\boldsymbol{\Sigma}}$ *is not positive semi-definite, we project* $\widetilde{\boldsymbol{\Sigma}}$ *onto its closest positive semi-definite matrix by solving*

$$\boldsymbol{\Sigma} = \underset{\boldsymbol{S} \succeq 0}{argmin} \ \|\boldsymbol{S} - \widetilde{\boldsymbol{\Sigma}}\|_F^2. \tag{9}$$

The solution to (9) *is the truncated eigen-decomposition where negative eigenvalues of* $\widetilde{\boldsymbol{\Sigma}}$ *are set to 0.*

D. Optimal MSE Lower Bound

We derive the MSE lower bound of (P_1). To do so, we consider a relaxed optimization by removing the non-negativity constraint:

$$\begin{aligned} &\underset{\boldsymbol{w}}{\text{minimize}} \ \ \boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w} \\ &\text{subject to} \ \ \boldsymbol{w}^T\mathbf{1} = 1. \end{aligned} \tag{P_2}$$

Clearly, the feasible set of (P_2) includes the feasible set of (P_1), and so the MSE obtained by solving (P_2) must be a lower bound of the MSE obtained by solving (P_1). More precisely, if we let $\widehat{\boldsymbol{w}}$ be the optimal weight vector obtained by (P_1), and $\boldsymbol{w}^*$ be that obtained by (P_2), then

$$\mathbb{E}\left[\left\|\widehat{\boldsymbol{Z}}\widehat{\boldsymbol{w}} - z\right\|^2\right] \geq \mathbb{E}\left[\left\|\widehat{\boldsymbol{Z}}\boldsymbol{w}^* - z\right\|^2\right]. \tag{10}$$

Let us analyze the right hand side of (10). The optimization in (P_2) is a standard linear equality constrained quadratic minimization. Closed-form solution can be derived via the standard Lagrangian approach by defining:

$$\mathcal{L}(\boldsymbol{w}, \lambda) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w} - \lambda(\boldsymbol{w}^T\mathbf{1} - 1). \tag{11}$$

The first order KKT conditions state that

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = 0, \quad \boldsymbol{w}^T\mathbf{1} = 1,$$

where the first condition is equivalent to

$$\boldsymbol{\Sigma}\boldsymbol{w} - \lambda\mathbf{1} = 0, \quad \text{or } \boldsymbol{w} = \lambda\boldsymbol{\Sigma}^\dagger\mathbf{1}, \tag{12}$$

where $\boldsymbol{\Sigma}^\dagger$ denotes the pseudo-inverse of a symmetric positive semi-definite matrix $\boldsymbol{\Sigma}$. If $\boldsymbol{\Sigma}$ is positive definite, then $\boldsymbol{\Sigma}^\dagger = \boldsymbol{\Sigma}^{-1}$ and (12) can be written as $\boldsymbol{w} = \lambda\boldsymbol{\Sigma}^{-1}\mathbf{1}$. Substituting (12) into the constraint, we have that

$$\mathbf{1}^T\left(\lambda\boldsymbol{\Sigma}^\dagger\mathbf{1}\right) = 1 \quad \Rightarrow \quad \lambda = \frac{1}{\mathbf{1}^T\boldsymbol{\Sigma}^\dagger\mathbf{1}}. \tag{13}$$

Substituting (13) into (12), we prove the following.

Proposition 4: The solution to (P_2) *is given by*

$$\boldsymbol{w}^* = \frac{\boldsymbol{\Sigma}^\dagger\mathbf{1}}{\mathbf{1}^T\boldsymbol{\Sigma}^\dagger\mathbf{1}}, \tag{14}$$

where $\boldsymbol{\Sigma}^\dagger$ *denotes the pseudo-inverse of the symmetric positive semi-definite matrix* $\boldsymbol{\Sigma}$.

Given the optimal weight vector $\boldsymbol{w}^*$, we can determine the corresponding mean squared error:

$$\mathbb{E}\left[\left\|\widehat{\boldsymbol{Z}}\boldsymbol{w}^* - z\right\|^2\right] = (\boldsymbol{w}^*)^T\boldsymbol{\Sigma}\boldsymbol{w}^* = \frac{1}{\mathbf{1}^T\boldsymbol{\Sigma}^\dagger\mathbf{1}}. \tag{15}$$

Since the weight $\boldsymbol{w}^*$ provides a lower bound on the MSE, in particular if we consider a weight vector $\boldsymbol{e}_k = [0, \ldots, 1, \ldots, 0]^T$ (i.e., the k-th standard basis vector), we must have

$$\text{MSE}_k = \boldsymbol{e}_k^T \boldsymbol{\Sigma} \boldsymbol{e}_k \geq \widehat{\boldsymbol{w}}^T \boldsymbol{\Sigma} \widehat{\boldsymbol{w}} \geq \frac{1}{\mathbf{1}^T \boldsymbol{\Sigma}^\dagger \mathbf{1}}. \tag{16}$$

The first inequality holds because $\boldsymbol{e}_k$ is one of the feasible vectors of (P_1) but $\widehat{\boldsymbol{w}}$ is the optimal solution. The second inequality holds because $\boldsymbol{w}^*$ is a solution of (P_2). The result of (16) states that an optimally combined estimate using $\widehat{\boldsymbol{w}}$ has to be at least as good as any initial estimate.

Remark 2: The MSE lower bound result presented here is more general than the previous result by Odell et al. [27] which only considered $K = 2$. When $K = 2$, we have

$$w_1^* = \frac{\Sigma_{22} - \Sigma_{12}}{\Sigma_{11} + \Sigma_{22} - 2\Sigma_{12}}, \quad \textit{and } w_2^* = 1 - w_1^*, \tag{17}$$

which is the same as [27, eq. (2), Table 3]. [2]

E. Perturbation in $\boldsymbol{\Sigma}$

We conclude this section by discussing the perturbation issue when we use an estimated covariance matrix $\widetilde{\boldsymbol{\Sigma}}$ instead of $\boldsymbol{\Sigma}$. To facilitate the discussion, we define two weight vectors:

$$\widetilde{\boldsymbol{w}} = \underset{\boldsymbol{v} \in \Omega}{\text{argmin}} \ \boldsymbol{v}^T \widetilde{\boldsymbol{\Sigma}} \boldsymbol{v}, \quad \text{and } \boldsymbol{w} = \underset{\boldsymbol{v} \in \Omega}{\text{argmin}} \ \boldsymbol{v}^T \boldsymbol{\Sigma} \boldsymbol{v}. \tag{18}$$

That is, $\widetilde{\boldsymbol{w}}$ is the optimal weight vector found according to the estimated covariance matrix $\widetilde{\boldsymbol{\Sigma}}$, and $\boldsymbol{w}$ is the optimal weight vector found according to the true covariance matrix $\boldsymbol{\Sigma}$. Correspondingly, we define their combined estimates as

$$\widetilde{z} = \widehat{\boldsymbol{Z}} \widetilde{\boldsymbol{w}}, \quad \text{and } \widehat{z} = \widehat{\boldsymbol{Z}} \boldsymbol{w}. \tag{19}$$

The following proposition summarizes the perturbation result.

Proposition 5: Assume that $\widetilde{\boldsymbol{\Sigma}} \succ 0$ and $\boldsymbol{\Sigma} \succ 0$. Then,

$$\mathbb{E}\|\widetilde{z} - \widehat{z}\|^2 \leq \mathbb{E}\|\widehat{z} - z\|^2 (2\Delta + \Delta^2), \tag{20}$$

where

$$\Delta \stackrel{def}{=} \|\widetilde{\boldsymbol{\Sigma}} \boldsymbol{\Sigma}^{-1} - \widetilde{\boldsymbol{\Sigma}}^{-1} \boldsymbol{\Sigma}\|_2.$$

Proof: The proof is given in the Appendix. Our proof simplifies the multi-block concept of [28]. We also utilize the generalized Rayleigh quotient idea to obtain the bound. □

The implication of Proposition 5 can be seen from the two terms on the right hand side of (20). First, $\mathbb{E}\|\widehat{z} - z\|^2$ measures the bias between the oracle combination $\widehat{z}$ and the ground truth z. That it is an upper bound in (20) implies that the perturbed estimate is upper limited by the bias. The second term Δ measures the closeness between the oracle covariance $\boldsymbol{\Sigma}$ and the estimated covariance $\widetilde{\boldsymbol{\Sigma}}$. If $\boldsymbol{\Sigma} \widetilde{\boldsymbol{\Sigma}}^{-1} = \boldsymbol{I}$, then $\Delta = 0$ and so the perturbation is minimized. In practice, if $\widetilde{\boldsymbol{\Sigma}}$ can be estimated in n random trials and if $\boldsymbol{\Sigma} \widetilde{\boldsymbol{\Sigma}}_n^{-1} \stackrel{p}{\to} \boldsymbol{I}$ as $n \to \infty$, then we can also show that $\Delta \stackrel{p}{\to} 0$. (For example, use SURE on multiple noisy observations, if available.)

[2] In [27, Eq. (2), Table 3], there is a typo of the numerator which should be corrected as $m_{22} - m_{12}$.

III. MSE Estimator

The key to make (P_1) succeed is an accurate covariance matrix $\boldsymbol{\Sigma}$. Estimating the covariance matrix requires estimating the mean squared error (MSE). In this section we discuss a neural network solution.

A. Why Not SURE?

In image processing, perhaps the most popular approach to estimate MSE is the Stein's Unbiased Risk Estimator (SURE). (See [29], [42] for illustrations, [43] for a Monte-Carlo version, and [44] for a recent work using SURE in deep neural network.) As its name suggested, SURE is an unbiased estimator of the true MSE, i.e., the estimator will approach to the true MSE as the number of samples grows.

While SURE-based estimators work well in ideal situations, it also has many shortcomings:

- Large Variance. SURE only provide *average performance* guarantee. For Monte-Carlo SURE, there is another level of randomness due to the Monte-Carlo scheme. Therefore, given a single noisy image, SURE can be inaccurate, especially for non-linear denoises such as BM3D.
- Clipped Noise. SURE is designed to handle additive i.i.d. Gaussian noise. However, most real images are clipped to $[0, 1]^N$. Most neural network denoisers also clip the signal to stabilize training. If the observed image is clipped, then SURE will fail [45].
- Beyond Denoisers. While SURE is a good choice for image denoising problems, one has to re-derive the SURE equations for different forward models, e.g., deblurring or super-resolution. This severely limits the generality of the present optimal combination framework.

To illustrate the problems of SURE, we conduct two experiments comparing SURE and the proposed neural network approach. The task of the experiments is to denoise the `cameraman256` image, corrupted by i.i.d. Gaussian noise of different noise levels. In the first experiment, the i.i.d. Gaussian noise is unclipped so that the theory of SURE applies. The result of this experiment is shown in Figure 6(a). The average of SURE (over 100 random trials of different noise realizations) is very similar to the true MSE, something we expect from the theory. However, the variance of SURE is big; indeed very big. If we use SURE to construct a $\boldsymbol{\Sigma}$, the resulting $\boldsymbol{\Sigma}$ can be bad.

The second experiment modifies the i.i.d. Gaussian noise to clipped Gaussian so that the resulting signal is bounded to $[0, 1]$. We argue that the clipped noise is more realistic because no physical sensor can produce a signal level below 0 or beyond 1. When the noise is clipped, the symmetry of Gaussian distribution is destroyed and the clipping is signal dependent. As a result, the MSE predicted by SURE is significantly off from the theory. Figure 6(b) illustrates the result. SURE produces a completely opposite trend of the MSE whereas the NN produces a more reasonable estimate.

B. Neural Network MSE Estimator

Our proposed solution is a deep neural network MSE estimator. Using deep neural networks for image quality

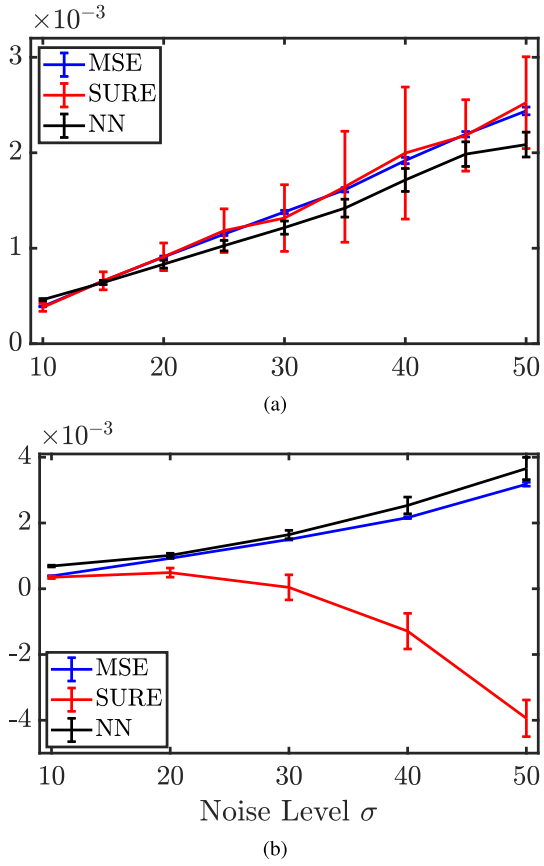


Fig. 6. (a) Unclipped and (b) clipped noise examples. Compare SURE and the proposed neural network (NN) on estimating the MSE. In this experiment, we use BM3D to denoise the `cameraman` image. The noise level changes from $\sigma = 10$ to $\sigma = 50$. The observed images are clipped to $[0, 1]^N$. The error bars are computed using 50 random trials of the i.i.d. Gaussian noise realizations. Dotted lines indicate the max and min of the realizations.

assessment is an active research topic [46]–[50]. However, the existing neural network based image quality assessment methods are tailored to predict the human visual system responses when presenting an image to a user. A pure MSE estimator is not common. To the best of our knowledge, the only existing MSE estimator is [48]. However, the MSE estimator in [48] is used to quantify *noisy* images, i.e., the amount of noise. An MSE estimator for *denoised* images does not currently exist.

The proposed neural network based MSE estimator is shown in Figure 7. There are two unique features of the network. First, the input to the network is a pair of images $(\boldsymbol{y}, \widehat{\boldsymbol{z}}_k)$, i.e., the noisy observation and the k-th denoised image. Using both $\boldsymbol{y}$ and $\widehat{\boldsymbol{z}}_k$ is reminiscent to the SURE approach, as $\boldsymbol{y}$ provides noise statistics that cannot be obtained from $\widehat{\boldsymbol{z}}_k$ alone.

Second, instead of feeding the entire image into the network, we partition the image into non-overlapping patches of size 64×64. That is, if we denote the MSE of the i-th patch of the k-th denoiser as $\widetilde{\text{MSE}}_{k,i} \stackrel{\text{def}}{=} \widetilde{\text{MSE}}(\boldsymbol{y}_i, \widehat{\boldsymbol{z}}_{k,i})$, then the overall MSE of the k-th denoiser is

$$\widetilde{\text{MSE}}_k = \frac{1}{M} \sum_{i=1}^{M} \widetilde{\text{MSE}}(\boldsymbol{y}_i, \widehat{\boldsymbol{z}}_{k,i}),$$

where $\boldsymbol{y}_i$ is the i-th patch of $\boldsymbol{y}$, $\widehat{\boldsymbol{z}}_{k,i}$ is the i-th patch of $\widehat{\boldsymbol{z}}_k$, and M is the number of non-overlapping patches in the image. Partitioning the image into small patches reduces the breadth and depth of the neural network.

The network consists of 8 convolutional layers, 3 maxpool layers and 2 fully connected layers. The inputs to the network are the i-th noisy patch $\boldsymbol{y}_i$ and the i-th denoised patch $\widehat{\boldsymbol{z}}_{k,i}$ of the k-th denoiser. The patches separately pass through two convolutional layers, and then concatenate and pass over four convolutional layers. The convoultional layers use 3×3 kernels with zero-padding and the rectifier activation function (ReLU). We apply maxpool layer with 2×2 kernel every two convolutional layers. Fully connected layers use ReLU and dropout regularization of ratio 0.5. The cost function is the L_1-loss, defined as

$$L = |\text{MSE}_{k,i} - \widetilde{\text{MSE}}_{k,i}| \tag{21}$$

where $\text{MSE}_{k,i}$ is the true MSE of i-th block of the k-th denoiser. For implementation, we use ADAM optimizer [51] with learning rate $\alpha = 10^{-4}$.

The training data we use is the 300 Training and Validation images in BSD500. For each image, we randomly extract 32 patches of size 64×64 and generate 6 variations by flipping horizontally and vertically and rotating at 0°, 90°, 180° and 270°. The noise level is $\sigma \in [1, 60]$, with clipping to $[0, 1]^N$. To prepare denoised images for training the networks, we use five pre-trained REDNets [17] at noise levels $\widehat{\sigma} = 10, 20, 30, 40, 50$. Therefore, for every noisy input we generate multiple denoised images, and every denoised image forms an input-output pair with the ground truth MSE. We train the MSE estimator network with 100 epochs for around 7 hours.

C. Comparison With SSDA

Readers familiar with the image denoising literature may ask about the difference between the proposed method and the AMC-SSDA method by Agostinelli *et al.* [32] (or SSDA in short). The SSDA method is an end-to-end neural network for denoising images of different noise types, e.g., salt-pepper, Gaussian, and Poisson. We are not interested in this problem because it is less common to have an image denoising problem where the noise type is totally blind. In contrast, it is more likely to have multiple denoisers for different noise levels (Section V-A), different image classes (Section V-D), and different denoiser types (Section V-E).

There are other differences. First, the SSDA has a set of fixed neural network denoisers. In contrast, CsNet can support *any* initial denoisers. Second, the weight prediction of the SSDA is done using a neural network which does not have optimality guarantee. CsNet, however, is optimal in the MMSE sense. Additionally, CsNet estimates the MSE (which is a scalar) from an image. This is easier than estimating the weight vector in SSDA. Third, CsNet can be generalized to other estimation problems such as deblurring and super-resolution. SSDA, however, has limited generalization capability because the initial estimators are limited to SSDA.

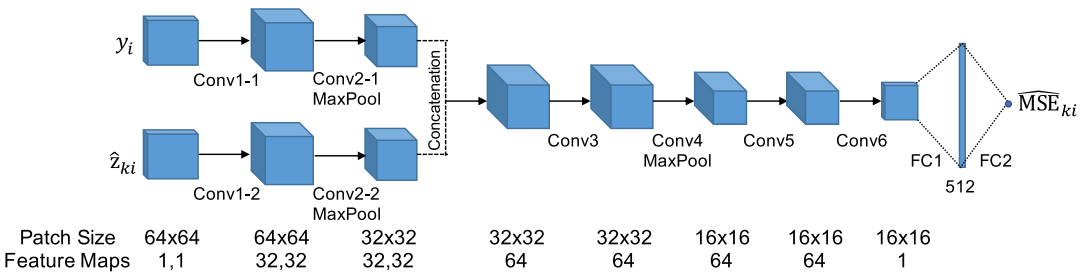


Fig. 7. Network structure of a proposed MSE estimator.

IV. Booster Network

In our proposed CsNet, besides the convex optimization algorithm and the MSE estimator, there is a third component known as the booster. The booster is used to improve the combined estimates by enhancing the contrast and to recover lost details. To provide readers a quick preview of the booster, we show a few examples in Figure 9.

A. What Is a Booster?

The concept of boosting can be traced back to at least the 70's, when Tukey [52] proposed a "twicing procedure". In machine learning, the same concept was studied by Bühlmann and Yu [53]. The essential step of boosting is simple: Given a current estimate $\widehat{\boldsymbol{z}}^{(t)}$ and the observation $\boldsymbol{y}$, we construct a mapping $\mathcal{B} : \mathbb{R}^N \to \mathbb{R}^N$ (usually another denoising algorithm), and then define the next estimate $\widehat{\boldsymbol{z}}^{(t+1)}$ in terms of $\widehat{\boldsymbol{z}}^{(t)}$, $\boldsymbol{y}$ and $\mathcal{B}$ with the goal to improve the MSE. In Tukey's "twicing", the relationship between $\widehat{\boldsymbol{z}}^{(t)}$ and $\widehat{\boldsymbol{z}}^{(t+1)}$ is

$$\widehat{\boldsymbol{z}}^{(t+1)} = \mathcal{B}(\boldsymbol{y} - \widehat{\boldsymbol{z}}^{(t)}) + \widehat{\boldsymbol{z}}^{(t)}. \tag{22}$$

Thus, if $\mathcal{B}$ is a denoiser, then $\mathcal{B}(\boldsymbol{y} - \widehat{\boldsymbol{z}}^{(t)})$ is the filtered version of the residue. As shown in [54], MSE is not monotonically decreasing as $t \to \infty$ because of the bias-variance trade-off. However, with proper monitoring such as cross-validation, MSE can be minimized by stopping the boosting procedure before saturation. (See additional discussion for the image denoising problem in [55].)

In the image denoising literature, the above idea of boosting has been studied in multiple places such as [54]–[56]. There are several variations, e.g., Osher's iterative regularization [57], and Romano and Elad's SOS [58]. In all these boosting methods, the idea is the take the noisy input and the estimate $\widehat{\boldsymbol{z}}^{(t)}$ to recursively update the estimate. Table I shows a comparison of different denoising boosters.

B. Deep Learning Based Booster

Our proposed neural network booster is motivated by the above examples of classical boosters. The specific network architecture is shown in Figure 8. Instead of using a deterministic function $\mathcal{B}$, we use a multi-layer neural network as the building block of the booster. We then cascade the building blocks to form the overall booster.

TABLE I

Different Denoising Boosters in the Literature. Our Proposed Method Generalizes the Classical BOOSTERS by Replacing $\mathcal{B}$ With Deep Neural Networks $\mathcal{B}_t$

Method	Idea
Twicing [52], [53]	$\widehat{\boldsymbol{z}}^{(t+1)} = \mathcal{B}(\boldsymbol{y} - \widehat{\boldsymbol{z}}^{(t)}) + \widehat{\boldsymbol{z}}^{(t)}$
Osher et al. [57]	$\widehat{\boldsymbol{z}}^{(t+1)} = \mathcal{B}\left(\boldsymbol{y} + \sum_{i=1}^{t}(\boldsymbol{y} - \widehat{\boldsymbol{z}}^{(t)})\right)$
Charest-Milanfar [54]	$\widehat{\boldsymbol{z}}^{(t+1)} = \boldsymbol{y} + \left(\widehat{\boldsymbol{z}}^{(t)} - \mathcal{B}(\widehat{\boldsymbol{z}}^{(t)})\right)$
Talebi-Milanfar [55]	$\widehat{\boldsymbol{z}}^{(t+1)} = \mathcal{B}(\boldsymbol{y} - \widehat{\boldsymbol{z}}^{(t)}) + \widehat{\boldsymbol{z}}^{(t)}$
Romano-Elad [58]	$\widehat{\boldsymbol{z}}^{(t+1)} = \mathcal{B}(\boldsymbol{y} + \widehat{\boldsymbol{z}}^{(t)}) - \widehat{\boldsymbol{z}}^{(t)}$
Proposed	$\widehat{\boldsymbol{z}}^{(t+1)} = \mathcal{B}_t(\boldsymbol{y}, \widehat{\boldsymbol{z}}^{(t)}) + \widehat{\boldsymbol{z}}^{(t)}$

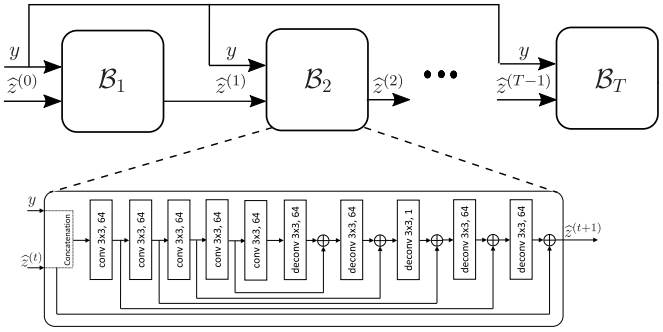


Fig. 8. Network structure of the proposed booster network. The network contains 5 convolutional layers followed by 5 deconvolutional layers. Convolutional and deconvolutional layers consist of residual neural network blocks. Skip connections are used to enforce symmetry of the network. This network is repeated five times, i.e., $T = 5$.

Referring to Figure 8, if we denote the t-th building block as $\mathcal{B}_t$, then the input-output relationship of $\mathcal{B}_t$ is

$$\widehat{\boldsymbol{z}}^{(t+1)} = \mathcal{B}_t(\boldsymbol{y}, \widehat{\boldsymbol{z}}^{(t)}) + \widehat{\boldsymbol{z}}^{(t)}. \tag{23}$$

Clearly, (23) is a generalization of (22) as $\mathcal{B}_t$ now becomes a nonlinear mapping trained from the data. Also, when cascading a sequence $\{\mathcal{B}_t\}$, we generalize (22) by allowing each $\mathcal{B}_t$ to have its own network weights.

The architecture of the t-th building block $\mathcal{B}_t$ consists of 5 convolutional layers followed by 5 deconvolutional layers, each using kernels of size 3×3. The input to the network is the pair $(\boldsymbol{y}, \widehat{\boldsymbol{z}}^{(t)})$, which is concatenated to form a common input. The convolutional layers are used to smooth out the noisy input $\boldsymbol{y}$, whereas the deconvolutional layers are used to

Fig. 9. Examples showing the effectiveness of the booster in improving the details and contrast of the combined results. See Section V-E for experiment details.

recover the sharp details. Skip connections are used to ensure signal is not attenuated as it passes through the layers. Note that we purposely add a skip connection from the input $\widehat{z}^{(t)}$ to the output $\widehat{z}^{(t+1)}$ to mimic the addition in (22). We cascade $\mathcal{B}_t$ for $t = 1, \ldots, T$, where T is typically small ($T = 5$).

To train a booster, we feed the booster network with linearly combined estimates and the ground truths. The initial denoisers are the REDNets at different noise levels. The training data we use is the 300 train and validation images in BSD500. We extract 32 patches of size 64×64 from each training dataset. For each patch we generate 6 variations by flipping horizontally and vertically and rotating at 0°, 90°, 180° and 270°. The cost function we use in training the booster network is the standard L_1-loss:

$$L = \left\| z - \widehat{z}^{(T)} \right\|_1 \tag{24}$$

where $\text{MSE}_{k,i}$ is the true MSE of i-th block of the k-th denoiser. During the training, we use ADAM optimizer with learning rate 10^{-4}. We trained booster network with 100 epochs for 12 hours.

C. Performance of Booster

The effectiveness of the booster can be seen in Figure 9, where we show a few examples taken from the BSD500 dataset. In this example, we consider a neural network denoiser trained at five different noise levels (See Section V-E for experiment details).

As we see in Figure 9, the booster is doing particularly well for two types of improvements. The first type of improvement is the recovery of the fine details. For example, in the Swam image we can recover the lines on the feather; in the House image we can recover branches of the tree. These are also reflected in the PSNR. The second type of improvement is the contrast enhancement. For example, before boosting the House image we see that the background sky has a gray-ish intensity. However, after boosting the background sky has a brighter background.

V. Experiments

We build our neural networks using Tensorflow and run on Intel(R) Core(TM) i5-4690K CPU 3.50GHz with an Nvidia Titan-X GPU, except DnCNN which is downloaded from the author's website.[3]

A. Experiment 1: Noise-Level Mismatch

Our first experiment is to evaluate CsNet for the case of noise-level mismatch. We consider two types of initial denoisers: DnCNN [19] and REDNet [17]. For each denoiser type, we use 300 training and validation images in BSD500 to train five initial denoisers $\mathcal{D}_1, \ldots, \mathcal{D}_5$. The denoising strength is set as one of the values $\widehat{\sigma} = 10, 20, 30, 40$, and 50. When testing, we use a noise level of $\sigma \in [10, 50]$. In this experiment, the noise is unclipped i.i.d. Gaussian.

The result of this experiment is shown in Table II and Figure 10. Table II shows the comparison with REDNet as initial denoisers, whereas Figure 10 shows a visual comparison of an image in the BSD500 dataset. We can make a few observations here:

- **General Performance**. For each σ, the best performing REDNet is the one with $\widehat{\sigma}$ right above σ. This result is consistent with the suggestion made by Zhang *et al.* [19]. However, the combination (before boosting) is able to improve the performance by an average of 0.3dB for noise levels that are originally not trained for, i.e., $\sigma = 15, 25, 35, 45$. For noise levels that are originally in the training set, i.e., $\sigma = 10, 20, 30, 40, 50$, the improvement is marginal.
- **Effect of Boosting**. If the actual noise level is unseen by the denoiser, e.g., $\sigma = 15$, the PSNR gain due to the booster is significant. For noise levels that have been observed, e.g., $\sigma = 20$, the gain is marginal. The reason

[3]Note that the original REDNet in [17] was implemented in Caffe, and the network was trained using patches of 50×50. We implemented REDNet on Tensorflow with patch size 64×64. On BSD200 dataset, our implementation shows better PSNR than the original REDNet.

TABLE II

EXPERIMENT 1A: NOISE-LEVEL MISMATCH FOR **UNCLIPPED NOISE**, WHERE NOISE IS I.I.D. GAUSSIAN *Without* CLIPPING THE SIGNAL TO [0, 1]. THE AVERAGE PSNRS OF REDNET ($\widehat{\sigma} = 10, 20, 30, 40, 50$), BLIND REDNET WITH 50 LAYERS AND CSNET ON 200 TEST IMAGES FROM BSD500. IN THIS FIGURE, "EST" AND "ORACLE" REFER TO ESTIMATED MSE AND THE ORACLE MSE, RESPECTIVELY

	REDNet ($\widehat{\sigma} = 10$)	REDNet ($\widehat{\sigma} = 20$)	REDNet ($\widehat{\sigma} = 30$)	REDNet ($\widehat{\sigma} = 40$)	REDNet ($\widehat{\sigma} = 50$)	Before Booster (est)	After Booster (est)	Before Booster (oracle)	After Booster (oracle)
$\sigma = 10$	**34.1705**	30.7509	28.2515	27.0308	25.9679	**34.1438**	33.9859	**34.1747**	33.9913
$\sigma = 15$	28.2492	**30.8902**	28.3384	27.0760	25.9920	31.4585	**31.7896**	31.4729	**31.7905**
$\sigma = 20$	24.1948	**30.4820**	28.4766	27.1502	26.0329	30.4768	**30.4805**	30.4931	**30.4888**
$\sigma = 25$	21.6813	26.6475	**28.6138**	27.2381	26.0826	29.0650	**29.2997**	29.0723	**29.3038**
$\sigma = 30$	19.8598	22.9125	**28.5231**	27.3544	26.1494	28.5199	**28.5494**	28.5323	**28.5571**
$\sigma = 35$	18.4271	20.5155	26.5631	**27.4453**	26.2322	27.7247	**27.8402**	27.7352	**27.8460**
$\sigma = 40$	17.2471	18.8398	23.2288	**27.2409**	26.3338	27.2387	**27.2781**	27.2542	**27.2887**
$\sigma = 45$	16.2479	17.5394	20.7749	25.3760	**26.4112**	26.6592	**26.7435**	26.6722	**26.7609**
$\sigma = 50$	15.3815	16.4471	18.9488	22.5099	**26.3197**	**26.3191**	26.3145	**26.3250**	26.3227

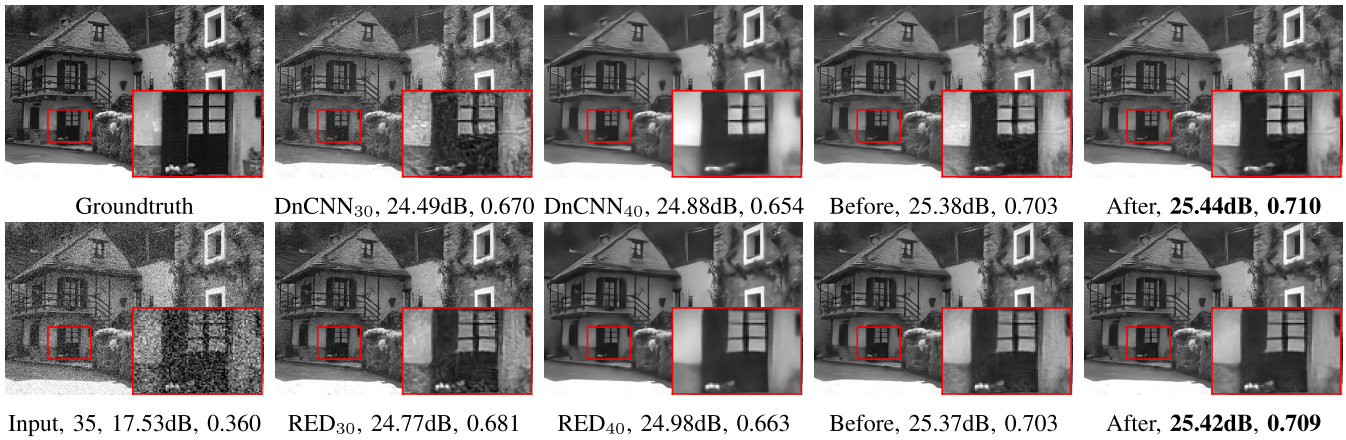


Fig. 10. Experiment 1: Noise-level mismatch for image `House` (size 321×481) from BSD500. The actual noise level is $\sigma = 35$. Top row: use DnCNN as initial denoisers; Bottom row: use REDNet as initial denoisers. Reported are the PSNR and SSIM values. In this figure, "before" and "after" refer to the result before and after applying the booster.

is that the booster has less room to improve when the denoised image is already good. This is consistent to the results reported in the boosting literature [58]. We also observe that for noise levels $\sigma = 10$ and $\sigma = 50$ there is a minor drop in the booster. This is because the booster is itself an estimator. When handling a wide range of noise levels, the network is only able to maximize the performance on the average case. For the extreme cases, there is a fundamental limitation which prevents the booster from being able to produce consistently good results. The same finding holds for other blind deep neural network denoisers, e.g., [19], which has worse performance for extreme low-noise and high-noise cases.

- **Oracle VS Estimate**. The difference between the oracle MSE and the estimated MSE is very small. Here, by oracle MSE we meant that the MSE is calculated from the ground truth. This will give us the best possible $\mathbf{\Sigma}$ when solving the convex optimization, and the PSNR can be regarded as the upper bound of any estimation method. As shown in the table, the performance of the MSE estimator is very similar to the oracle. This suggests that our neural network MSE estimator can reliably predict the MSE and hence facilitates the combination scheme.

B. Deeper Vanilla Network?

A natural question we can ask is that since we have five initial deep neural networks, is the performance gain due to the increased model capacity of the overall denoiser? To answer this question, we consider a blind denoiser of the same model capacity as the overall CsNet before boosting. Specifically, since we are using five REDNet-30 in the previous experiment, here we train a blind REDNet with 150 layers by repeating the structure of REDNet-30 five times. We call this the deep vanilla network.

The result of this experiment is shown in Table IV. The first two columns of this table show the unclipped noise performance using our proposed method. The third column is the vanilla 150-layer REDNet trained using noisy samples of noise level from 1 to 70. This is an advantageous setting, because the network is allowed to see samples of noise levels such as 15 or 35 which are not present in the five baseline REDNet-30's. The last column is another vanilla 150-layer

TABLE III

EXPERIMENT 1B: NOISE-LEVEL MISMATCH FOR THE **CLIPPED NOISE**, WHERE THE I.I.D. GAUSSIAN IS CLIPPED TO ENSURE THAT THE SIGNAL LIES IN [0, 1]. THE AVERAGE PSNRS OF REDNET ($\widehat{\sigma} = 10, 20, 30, 40, 50$), BLIND REDNET WITH 50 LAYERS AND CSNET ON 200 TEST IMAGES FROM BSD500. IN THIS FIGURE, "EST" AND "ORACLE" REFER TO ESTIMATED MSE AND THE ORACLE MSE, RESPECTIVELY

	REDNet ($\widehat{\sigma} = 10$)	REDNet ($\widehat{\sigma} = 20$)	REDNet ($\widehat{\sigma} = 30$)	REDNet ($\widehat{\sigma} = 40$)	REDNet ($\widehat{\sigma} = 50$)	Before Booster (est)	After Booster (est)	Before Booster (oracle)	After Booster (oracle)
$\sigma = 10$	**34.1428**	30.6934	28.2434	26.8287	25.8601	**34.0756**	33.9220	**34.1434**	33.9061
$\sigma = 15$	28.4337	**30.7544**	28.2961	26.8381	25.8532	31.3295	**31.7896**	31.3878	**31.8022**
$\sigma = 20$	24.4306	**30.3462**	28.3595	26.8382	25.8341	30.3121	**30.4621**	30.3516	**30.4763**
$\sigma = 25$	21.8383	26.9932	**28.4116**	26.8396	25.8065	28.8881	**29.3027**	28.9210	**29.3030**
$\sigma = 30$	19.9669	23.4285	**28.2041**	26.8316	25.7651	28.1983	**28.5163**	28.2213	**28.5225**
$\sigma = 35$	18.4955	21.0504	26.2027	**26.7998**	25.7074	27.2566	**27.7785**	27.2774	**27.7848**
$\sigma = 40$	17.2907	19.3423	23.2651	**26.6291**	25.6314	26.6547	**27.2147**	26.6777	**27.2208**
$\sigma = 45$	16.2759	18.0084	20.9738	**25.5692**	25.5244	25.9516	**26.6856**	25.9750	**26.6975**
$\sigma = 50$	15.3992	16.9077	19.2047	23.3792	**25.3426**	25.3940	**26.2533**	25.4284	**26.2612**

TABLE IV

CONSENSUSNET VS. DEEP VANILLA NETWORK. FOR THE DEEP VANILLA NETWORK, ONE REDNET IS TRAINED WITH $\widehat{\sigma} = 1, 2, \ldots, 70$ AND THE OTHER IS WITH $\widehat{\sigma} = 10, 20, \ldots, 50$ LIKE THE INITIAL DENOISERS

	Before Booster (est)	After Booster (est)	REDNet Blind 150 ($\widehat{\sigma}$=1,2,...,70)	REDNet Blind 150 ($\widehat{\sigma}$=10,20,...,50)
σ=10	**34.1438**	33.9859	33.8295	**33.9487**
σ=15	31.4585	**31.7896**	**31.7352**	30.2000
σ=20	30.4768	**30.4805**	30.3304	**30.3557**
σ=25	29.0650	**29.2997**	**29.2868**	28.1811
σ=30	28.5199	**28.5494**	28.4640	**28.4782**
σ=35	27.7247	**27.8402**	**27.7810**	27.1076
σ=40	27.2387	**27.2781**	**27.2084**	27.1945
σ=45	26.6592	**26.7435**	**26.7229**	25.0532
σ=50	**26.3191**	26.3145	**26.3013**	26.2898

REDNet, but trained using noise levels of {10, 20, 30, 40, 50}. This is more fair, as the network has the same training samples as the five baseline REDNet-30's. Both networks are trained with the same number of training examples.

As we observe from Table IV, the proposed combination scheme actually works better than the 150-layer REDNet. If we compare "before boosting" and the last column (the REDNet trained with the same set of samples as ours), the combination scheme produces significantly better performance in all cases. This suggests that the improvement is not due to the increased model capacity but the intrinsic power of the combination. If we allow the 150-layer REDNet to see the unseen examples (i.e., the third column), then the performance is worse than our "before boosting" for noise levels $\sigma = 10, 20, \ldots, 50$. For noise levels such as $15, 25, \ldots, 45$, the 150-layer REDNet is better than "before boosting". However, this is an unfair comparison because this REDNet is allowed to see images of those noise levels.

We also observe in some cases the weaker REDNet-150 (last column) performs better than the more powerful REDNet-150 (third column). These happens when $\sigma = 10, 20, 30$. One reason is that for the same amount of training examples, the more powerful REDNet distributes the training examples to all noise levels from 1 to 70, whereas the weaker REDNet only focuses on $10, 20, \ldots, 50$. This puts advantageous on the weaker REDNet-150 when it goes to those noise levels. In fact, even for $\sigma = 40$ and 50, the difference between the two REDNet's are marginal.

C. Clipped and Unclipped Noise

Since our proposed framework can be adapted to different types of noise (by training a different MSE estimator), here we demonstrate the performance of the proposed method on clipped and unclipped noise. To generate the clipped noisy image, we first add i.i.d. Gaussian noise to the image and clip the resulting image to the range [0, 1]. We argue that this is a more natural configuration, because most physical sensor have limited dynamic range.

The result of this experiment is shown in Table III. One thing to notice is that the REDNet's are still the same; They are re-trained using the clipped noise. As a result, their performance is worse than the unclipped version because of the training-testing mismatch. However, this deficiency of the initial denoiser brings out a useful feature of the proposed framework: Regardless of what the initial denoiser does, the proposed framework is able to pick the strongest denoiser and make improvements. If we look at Table III, besides the case of $\sigma = 10$, the proposed method is always better than the initial denoiser, despite the fact that the noise is clipped.

D. Experiment 2: Different Image Classes

The objective of this experiment is to evaluate the performance of CsNet when the initial denoisers are trained for different image classes. To this end, we fix the type of initial denoisers as REDNet, and train three different REDNets using three classes of images: `Flower`, `Face` and `Building`. We have experimented with other initial denoisers such as DnCNN, but the results are similar.

TABLE V

EXPERIMENT 2: DIFFERENT IMAGE CLASSES. CLASS-SPECIFIC REDNETS HAVE BETTER PERFORMANCE THAN BM3D, DNCNN (GENERIC) AND REDNET (GENERIC). CSNET SELECTS THE BEST CLASS. WE USE 10 IMAGES FROM IMAGENET FOR TESTING. THE LABELS "EST" AND "ORACLE" REFER TO ESTIMATED MSE AND THE ORACLE MSE, RESPECTIVELY

	REDNet (Building)	REDNet (Face)	REDNet (Flower)	Before Booster (est)	After Booster (est)	Before Booster (oracle)	After Booster (oracle)	BM3D	DnCNN (generic)	REDNet (generic)
Unclipped Noise										
Building	**30.6038**	29.1219	29.5430	30.3509	**30.9371**	30.6136	**30.9391**	29.5059	**30.0341**	29.9658
Face	30.5437	**30.7606**	30.7116	30.8047	**30.9923**	30.8907	**31.0569**	30.2397	30.6967	**30.7020**
Flower	31.2785	31.1325	**31.5428**	31.5788	**31.6035**	31.6009	**31.6103**	30.6088	**31.4211**	31.4105
Clipped Noise										
Building	**30.3962**	28.9529	29.3453	30.3303	**30.4095**	30.4020	**30.4749**	29.2986	29.7722	**29.7743**
Face	30.1871	**30.3889**	30.3443	30.4501	**30.7419**	30.5086	**30.7957**	29.9685	30.2813	**30.2896**
Flower	31.0875	30.9497	**31.3114**	31.3221	**31.5041**	31.3759	**31.5404**	30.4224	31.1534	**31.1752**

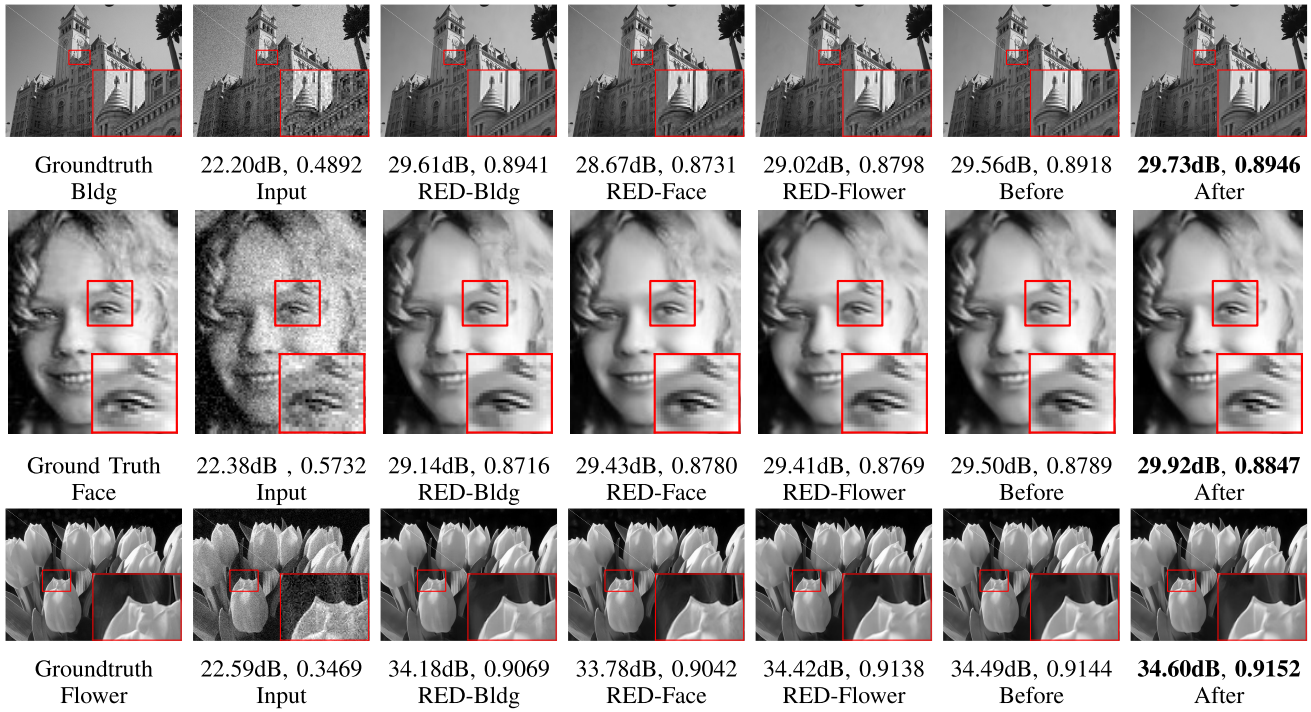


Fig. 11. Experiment 2: `Building`, `Face` and `Flower` classes. Testing images are from ImageNet. Reported are the PSNR and SSIM values. In this figure, "before" and "after" refer to the result before and after applying the booster.

To train the initial denoisers, we manually select 200 class-specific images for each class from the ImageNet [59]. We fix the noise level as $\sigma = 20$ to eliminate the complication of having uncertainty in both noise levels and image classes. Initial denoisers are trained with unclipped noise. We train two different MSE estimators, one for unclipped noise and one for clipped noise.

The result of this experiment is shown in Table V with a few representative examples in Figure 11. We observe that denoisers trained with generic database such as DnCNN (generic) and REDNet (generic) perform worse than class-specific denoisers. For example, in the `Building` image, DnCNN (generic) and REDNet (generic) attain 29.7722dB and 29.7743dB respectively in the clipped case. In contrast, REDNet-`Building` has a PSNR of 30.39dB, approximately 0.7dB above the REDNet (generic).

When using the proposed scheme, the "before boosting" result is already better than the initial denoiser's. This result holds for both clipped and unclipped, and all classes. Moreover, "before boosting" is better than all the generic denoisers, indicating the effectiveness of the convex optimization part. If we apply a booster, then the performance is boosted further.

E. Experiment 3: Different Denoiser Types

The objective of this experiment is to evaluate CsNet for different types of initial denoisers. To this end, we consider four denoisers running at specific noise levels $\widehat{\sigma}$ that match

TABLE VI

EXPERIMENT 3: DIFFERENT DENOISER TYPE. THE INITIAL DENOISERS ARE BM3D [2], DNCNN [19], REDNET [17], AND FFDNET [21]. WE USE 200 IMAGES FROM BSD500 FOR TESTING. IN THIS FIGURE, "BEFORE" AND "AFTER" REFER TO THE RESULT BEFORE AND AFTER APPLYING THE BOOSTER. THE LABELS "EST" AND "ORACLE" REFER TO ESTIMATED MSE AND THE ORACLE MSE, RESPECTIVELY

	BM3D [2]	DnCNN [19]	FFDNet [21]	REDNet [17]	Before Boost (est)	After Boost (est)	Before Boost (oracle)	After Boost (oracle)
Unclipped Noise								
$\sigma = 10$	33.6067	**34.1625**	34.0178	34.1619	**34.1813**	34.1678	**34.2147**	34.1906
$\sigma = 20$	29.8558	**30.4924**	30.4357	30.4755	30.5258	**30.5401**	**30.5559**	30.5554
$\sigma = 30$	27.9271	28.5286	**28.5458**	28.5209	28.5869	**28.6198**	28.6199	**28.6299**
$\sigma = 40$	26.5688	27.2202	**27.2845**	27.2393	27.2978	**27.3384**	27.3381	**27.3438**
$\sigma = 50$	25.7005	26.3159	**26.3675**	26.3249	26.3695	**26.4235**	**26.4226**	26.4223
Clipped Noise								
$\sigma = 10$	33.5628	34.1030	33.9434	**34.1216**	**34.1362**	33.8933	**34.1625**	33.9012
$\sigma = 20$	29.7309	30.3266	30.2683	**30.3378**	30.3672	**30.4846**	30.3994	**30.5076**
$\sigma = 30$	27.6804	28.1727	28.1846	**28.2007**	28.2282	**28.5211**	28.2764	**28.5529**
$\sigma = 40$	26.2208	26.6024	**26.6452**	26.6205	26.6788	**27.1906**	26.7187	**27.2108**
$\sigma = 50$	24.9885	25.3449	**25.3491**	25.3479	25.3952	**26.1573**	25.4354	**26.1766**

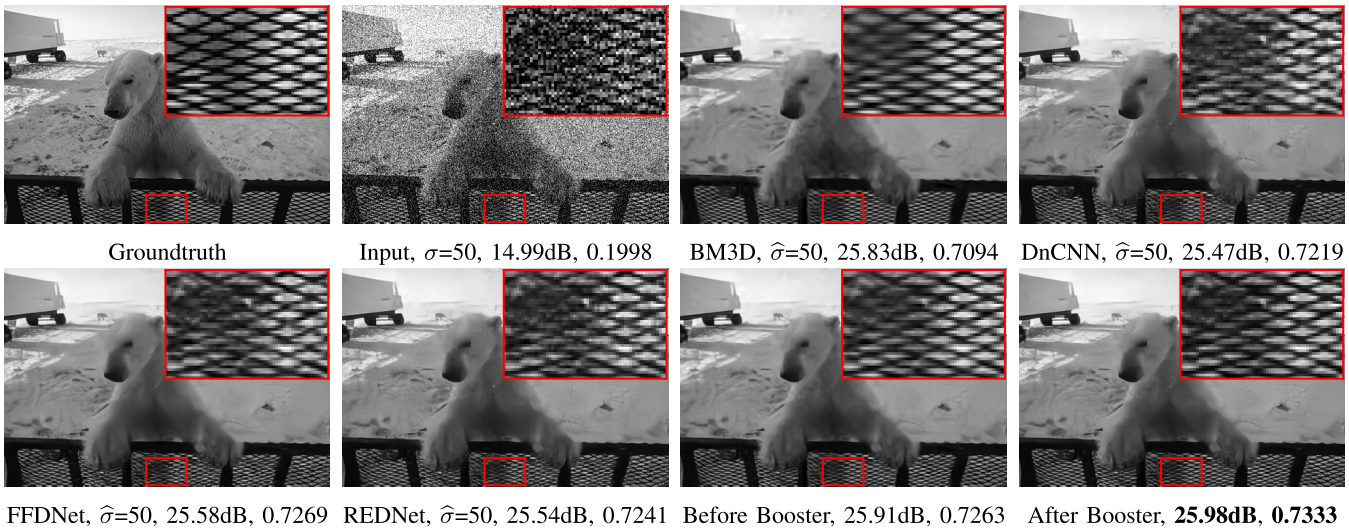


Fig. 12. Experiment 3: Different denoiser type. The initial denoisers are BM3D [2], DnCNN [19], REDNet [17], and FFDNet [21]. The testing image is `Bear` (size 321×481) from BSD500. Reported are the PSNR and SSIM values.

with the actual noise level σ. These denoisers are BM3D [2], DnCNN [19], REDNet [17] and FFDNet [21]. We use the original implementation by the authors for DnCNN and FFDNet, and build our own REDNet.

The result of this experiment is shown in Table VI. Among the four denoisers, FFDNet and REDNet have comparable performance at the top, followed by DnCNN and then BM3D. For the five noise levels we tested, CsNet consistently improves the performance. In particular, "before boosting" is always better than the initial denoiser. This means the convex optimization has effectively selected the best initial denoiser. The margin between the best initial denoiser and "before boosting" is small, because the denoisers have similar behavior and so the convex optimization solution is at one of the vertices of the constraint hyperplane. Figure 12 shows a visual comparison on the `Bear` image. In this image, BM3D actually performs better than DnCNN. The proposed CsNet can pick this best estimate (25.91dB), and boost the PSNR to 25.98dB.

F. Limitations and Extensions

The effectiveness of CsNet is dominated by the accuracy of the MSE estimate. The proposed neural network MSE estimator has a bias but a small variance. This is better than deterministic estimators such as SURE which has no bias but excessively large variance. However, if the noise statistics changes, we need to train a different MSE estimator.

If the images are large and complex, we can partition the image into sub-regions and use CsNet to handle each region separately. The bottleneck, again, is the accuracy in estimating the MSE. One resolution is to consider regularization in (P_1). Possible choices of regularization include forcing similar

weights for denoisers that are known to perform similarly. We leave the discussion of such regularization to future work.

Real noise of an image is significantly more complicated than i.i.d. Gaussian. Typical sources of noise include: photon shot noise, optical diffusion, minority carrier, thermal effect, dark current, circuit instability, and various nonlinear operations due to the image processing pipeline. When taking all these into account, a better noise model beyond Gaussian (and even mixed Poisson-Gaussian) is needed. Readers interested in this topic can consult, e.g., [60]–[62] for theory, and [63], [64] for some recent progress on algorithms. The current CsNet is not designed to handle this type of real noise. However, if one can show that real noise is a mixture of individual noises, then CsNet could potentially be a solution.

When training the neural networks we choose to use the L_1 metric, for it gives slightly better visual quality then the usual L_2 metric. We do not heavily tune this metric because it is not the focus of the paper. For readers who are concerned about the loss function, we refer to [65] for some recent empirical findings on the topic.

The advantage of CsNet relative to other class-aware neural network denoisers is that we allow combination of multiple denoisers. Typical class-aware denoisers, e.g., [18], [66], [67], rely on semantic classifiers to greedily select only one denoiser. As we demonstrated in Section V-D, a combination of the denoisers is better than the best of the individuals.

CsNet is a general framework for combining estimators. That is, one is not limited to applying CsNet to image denoising problems, although we use denoising as a demonstration. A straight forward extension of CsNet is to combine multiple deblurring algorithms, or to combine multiple image super-resolution algorithms. In complex imaging scenarios where no single method performs uniformly better than the others, CsNet offers a solution to integrate individual weak estimators.

VI. Conclusion

We present an optimal framework called the Consensus Neural Network (CsNet) to combine multiple weak image denoisers. CsNet consists of three major components. Starting with a set of initial image denoisers, CsNet first uses a novel deep neural network to estimate the MSE. The deep neural network is more robust than the traditional estimators such as SURE for estimating the MSE. Once the MSE is estimated, CsNet solves a convex optimization problem. The optimality of the CsNet is guaranteed by the convex formulation. Finally, the combined estimate is boosted using a new deep neural network image booster. Experimental results confirm the effectiveness of CsNet, where it shows superior performance compared to other state-of-the-art denoising algorithms on tasks including: overcoming noise level mismatch, combining denoisers for different image classes, and combining different denoiser types.

Appendix: Proofs

Proof of Proposition 5

First, we show that

$$\begin{aligned}\mathbb{E}\|\widetilde{\boldsymbol{z}}-\widehat{\boldsymbol{z}}\|^2 &\overset{\text{def}}{=} \mathbb{E}\|\widehat{\boldsymbol{Z}}\widetilde{\boldsymbol{w}}-\widehat{\boldsymbol{Z}}\boldsymbol{w}\|^2 = \mathbb{E}\|\widehat{\boldsymbol{Z}}\widetilde{\boldsymbol{w}}-\boldsymbol{z}+\boldsymbol{z}-\widehat{\boldsymbol{Z}}\boldsymbol{w}\|^2\\ &= \mathbb{E}\|(\widehat{\boldsymbol{Z}}\widetilde{\boldsymbol{w}}-\boldsymbol{Z}\widetilde{\boldsymbol{w}})-(\widehat{\boldsymbol{Z}}\boldsymbol{w}-\boldsymbol{Z}\boldsymbol{w})\|^2\\ &= \mathbb{E}\|(\widehat{\boldsymbol{Z}}-\boldsymbol{Z})(\widetilde{\boldsymbol{w}}-\boldsymbol{w})\|^2 = (\widetilde{\boldsymbol{w}}-\boldsymbol{w})^T\boldsymbol{\Sigma}(\widetilde{\boldsymbol{w}}-\boldsymbol{w}).\end{aligned}$$

The term $(\widetilde{\boldsymbol{w}}-\boldsymbol{w})^T\boldsymbol{\Sigma}(\widetilde{\boldsymbol{w}}-\boldsymbol{w})$ can be upper bounded by

$$\begin{aligned}(\widetilde{\boldsymbol{w}}-\boldsymbol{w})^T\boldsymbol{\Sigma}(\widetilde{\boldsymbol{w}}-\boldsymbol{w}) &= \widetilde{\boldsymbol{w}}^T\boldsymbol{\Sigma}\widetilde{\boldsymbol{w}}-\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}-2(\widetilde{\boldsymbol{w}}-\boldsymbol{w})^T\boldsymbol{\Sigma}\boldsymbol{w}\\ &\le \widetilde{\boldsymbol{w}}^T\boldsymbol{\Sigma}\widetilde{\boldsymbol{w}}-\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}.\end{aligned}$$

The last inequality holds because the function $f(\boldsymbol{w})=\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}$ attains its first order optimality at $\boldsymbol{w}$ when

$$\nabla f(\boldsymbol{w})^T(\widetilde{\boldsymbol{w}}-\boldsymbol{w})\ge 0.$$

Therefore,

$$\begin{aligned}&\widetilde{\boldsymbol{w}}^T\boldsymbol{\Sigma}\widetilde{\boldsymbol{w}}-\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}\\ &\quad= \widetilde{\boldsymbol{w}}^T\boldsymbol{\Sigma}\widetilde{\boldsymbol{w}}-\widetilde{\boldsymbol{w}}^T\widetilde{\boldsymbol{\Sigma}}\widetilde{\boldsymbol{w}}+\widetilde{\boldsymbol{w}}^T\widetilde{\boldsymbol{\Sigma}}\widetilde{\boldsymbol{w}}-\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}\\ &\quad\le \widetilde{\boldsymbol{w}}^T\boldsymbol{\Sigma}\widetilde{\boldsymbol{w}}-\widetilde{\boldsymbol{w}}^T\widetilde{\boldsymbol{\Sigma}}\widetilde{\boldsymbol{w}}+\boldsymbol{w}^T\widetilde{\boldsymbol{\Sigma}}\boldsymbol{w}-\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}\\ &\quad= \widetilde{\boldsymbol{w}}^T\widetilde{\boldsymbol{\Sigma}}\widetilde{\boldsymbol{w}}\left(\frac{\widetilde{\boldsymbol{w}}^T\boldsymbol{\Sigma}\widetilde{\boldsymbol{w}}}{\widetilde{\boldsymbol{w}}^T\widetilde{\boldsymbol{\Sigma}}\widetilde{\boldsymbol{w}}}-1\right)+\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}\left(\frac{\boldsymbol{w}^T\widetilde{\boldsymbol{\Sigma}}\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}}-1\right)\\ &\quad\le (\widetilde{\boldsymbol{w}}^T\widetilde{\boldsymbol{\Sigma}}\widetilde{\boldsymbol{w}}+\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w})\delta,\end{aligned}$$

where

$$\delta=\max\left(\left|\frac{\widetilde{\boldsymbol{w}}^T\boldsymbol{\Sigma}\widetilde{\boldsymbol{w}}}{\widetilde{\boldsymbol{w}}^T\widetilde{\boldsymbol{\Sigma}}\widetilde{\boldsymbol{w}}}-1\right|,\left|\frac{\boldsymbol{w}^T\widetilde{\boldsymbol{\Sigma}}\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}}-1\right|\right) \tag{25}$$

We can also show that

$$\boldsymbol{w}^T\widetilde{\boldsymbol{\Sigma}}\boldsymbol{w}\le\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}(1+\delta)$$

Continue the calculation, we have

$$\begin{aligned}(\widetilde{\boldsymbol{w}}^T\widetilde{\boldsymbol{\Sigma}}\widetilde{\boldsymbol{w}}+\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w})\delta &\le (\boldsymbol{w}^T\widetilde{\boldsymbol{\Sigma}}\boldsymbol{w}+\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w})\delta\\ &\le (\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w})(2\delta+\delta^2)\end{aligned}$$

This implies that

$$\mathbb{E}\|\widetilde{\boldsymbol{z}}-\widehat{\boldsymbol{z}}\|^2\le\mathbb{E}\|\widehat{\boldsymbol{z}}-\boldsymbol{z}\|^2(2\delta+\delta^2).$$

It remains to derive an upper bound on δ. To this end, we consider the generalized Rayleigh quotient of two positive definite matrices $\boldsymbol{A}$ and $\boldsymbol{B}$. It is known that [68]

$$\max_{\boldsymbol{w}\ne 0}\frac{\boldsymbol{w}^T\boldsymbol{A}\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{B}\boldsymbol{w}}=\lambda_{\max}\left(\boldsymbol{B}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{B}^{-\frac{1}{2}}\right).$$

Therefore,

$$\begin{aligned}\left|\frac{\boldsymbol{w}^T\widetilde{\boldsymbol{\Sigma}}\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}}-1\right| &\le \max_{\boldsymbol{w}\ne 0}\left|\frac{\boldsymbol{w}^T\widetilde{\boldsymbol{\Sigma}}\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}}-1\right| = \max_{\boldsymbol{w}\ne 0}\left|\frac{\boldsymbol{w}^T(\widetilde{\boldsymbol{\Sigma}}-\boldsymbol{\Sigma})\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}}\right|\\ &= \max_i\left|\lambda_i\left(\boldsymbol{\Sigma}^{-\frac{1}{2}}(\widetilde{\boldsymbol{\Sigma}}-\boldsymbol{\Sigma})\boldsymbol{\Sigma}^{-\frac{1}{2}}\right)\right|,\end{aligned}$$

where $\lambda_i(\boldsymbol{A})$ denotes the i-th eigen-value of the matrix $\boldsymbol{A}$. With some additional algebra we can show that

$$\begin{aligned}&\max_i\left|\lambda_i\left(\boldsymbol{\Sigma}^{-\frac{1}{2}}(\widetilde{\boldsymbol{\Sigma}}-\boldsymbol{\Sigma})\boldsymbol{\Sigma}^{-\frac{1}{2}}\right)\right|\\ &\quad= \max_i\left|1-\lambda_i\left(\boldsymbol{\Sigma}^{-\frac{1}{2}}\widetilde{\boldsymbol{\Sigma}}\boldsymbol{\Sigma}^{-\frac{1}{2}}\right)\right|\\ &\quad\overset{(a)}{=} \max_i\left|1-\lambda_i\left(\boldsymbol{\Sigma}^{-1}\widetilde{\boldsymbol{\Sigma}}\right)\right|\\ &\quad\overset{(b)}{\le} \max_i\left|\frac{1}{\lambda_i\left(\boldsymbol{\Sigma}^{-1}\widetilde{\boldsymbol{\Sigma}}\right)}-\lambda_i\left(\boldsymbol{\Sigma}^{-1}\widetilde{\boldsymbol{\Sigma}}\right)\right|,\end{aligned}$$

where (a) holds because of Lemma 1, and (b) holds because for any $t \geq 0$, $|1-t| \leq |t-\frac{1}{t}|$. By recalling the definition of the matrix operator norm, we have that

$$\left| \frac{\boldsymbol{w}^T \widetilde{\boldsymbol{\Sigma}} \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{\Sigma} \boldsymbol{w}} - 1 \right| \leq \left\| \boldsymbol{\Sigma} \widetilde{\boldsymbol{\Sigma}}^{-1} - \boldsymbol{\Sigma}^{-1} \widetilde{\boldsymbol{\Sigma}} \right\|_2 \stackrel{\text{def}}{=} \Delta .$$

Substituting this result into (25), and by symmetry, we complete the proof.

Lemma 1: Consider two matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n \times n}$ where $\boldsymbol{AB}$ and $\boldsymbol{BA}$ are diagonalizable. If λ is an eigen-value of $\boldsymbol{AB}$, then λ is also an eigen-value of $\boldsymbol{BA}$.

Proof: Let $\boldsymbol{v} \in \mathbb{R}^n$ be an eigen-vector of $\boldsymbol{AB}$, i.e.,

$$\boldsymbol{ABv} = \lambda \boldsymbol{v}.$$

Then, multiplying both sides by $\boldsymbol{B}$ yields

$$\boldsymbol{BA}\left(\boldsymbol{Bv}\right) = \lambda \left(\boldsymbol{Bv}\right).$$

Hence, λ is an eigen-value of $\boldsymbol{BA}$, with the corresponding eigen-vector $\boldsymbol{Bv}$. □

ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for very valuable feedback which significantly improves the paper. They also thank Nvidia for donating the Titan-X GPU for this work.

REFERENCES

[1] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 60–65.

[2] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[3] A. Beck and M. Teboulle, "Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems," *IEEE Trans. Image Process.*, vol. 18, no. 11, pp. 2419–2434, Nov. 2009.

[4] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," in *Proc. IEEE Int Conf Comput. Vis. (ICCV)*, Nov. 2011, pp. 479–486.

[5] S. H. Chan, R. Khoshabeh, K. B. Gibson, P. E. Gill, and T. Q. Nguyen, "An augmented Lagrangian method for total variation video restoration," *IEEE Trans. Image Process.*, vol. 20, no. 11, pp. 3097–3111, Nov. 2011.

[6] S. Gu, L. Zhang, W. Zuo, and X. Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, pp. 2862–2869.

[7] S. H. Chan, T. Zickler, and Y. M. Lu, "Monte Carlo non-local means: Random sampling for large-scale image filtering," *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3711–3725, Aug. 2014.

[8] Y. Chi and S. H. Chan, "Fast and robust recursive filter for image denoising," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 1708–1712.

[9] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Process.*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006.

[10] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 2272–2279.

[11] S. Roth and M. J. Black, "Fields of experts," *Int. J. Comput Vis.*, vol. 82, no. 2, pp. 205–229, Apr. 2009.

[12] H. C. Burger, C. J. Schuler, and S. Harmeling, "Image denoising: Can plain neural networks compete with BM3D?" in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 2392–2399.

[13] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 341–349.

[14] W. Dong, L. Zhang, G. Shi, and X. Li, "Nonlocally centralized sparse representation for image restoration," *IEEE Trans. Image Process.*, vol. 22, no. 4, pp. 1620–1630, Apr. 2013.

[15] W. Dong, G. Shi, and X. Li, "Nonlocal image restoration with bilateral variance estimation: A low-rank approach," *IEEE Trans. Image Process.*, vol. 22, no. 2, pp. 700–711, Feb. 2013.

[16] J. Xu, L. Zhang, W. Zuo, D. Zhang, and X. Feng, "Patch group based nonlocal self-similarity prior learning for image denoising," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 244–252.

[17] X. Mao, C. Shen, and Y. Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 2802–2810.

[18] T. Remez, O. Litany, R. Giryes, and A. M. Bronstein, "Deep class-aware image denoising," in *Proc. Int. Conf. Sampling Theory Appl. (SampTA)*, Sep. 2017, pp. 138–142.

[19] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.

[20] S. Lefkimmiatis, "Non-local color image denoising with convolutional neural networks," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3587–3596.

[21] K. Zhang, W. Zuo, and L. Zhang. (Oct. 2017). "FFDNet: Toward a fast and flexible solution for CNN based image denoising." [Online]. Available: https://arxiv.org/abs/1710.04026, Oct. 2017.

[22] J. Xu, L. Zhang, and D. Zhang, "A trilateral weighted sparse coding scheme for real-world image denoising," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 20–36.

[23] F. A. Graybill and R. B. Deal, "Combining unbiased estimators," *Biometrics*, vol. 15, no. 4, pp. 543–550, Dec. 1959.

[24] E. Samuel-Cahn, "Combining unbiased estimators," *Amer. Statist.*, vol. 48, no. 1, pp. 34–36, Feb. 1994.

[25] D. B. Rubin and S. Weisberg, "The variance of a linear combination of independent estimators using estimated weights," *Biometrika*, vol. 62, no. 3, pp. 708–709, Dec. 1975.

[26] T. Keller and I. Olkin, "Combining correlated unbiased estimators of the mean of a normal distribution," Lecture Notes—Monograph Series, vol. 45. Beachwood, OH, USA: Institute of Mathematical Statistics, 2004, pp. 218–227.

[27] P. L. Odell, D. Dorsett, D. Young, and J. Igwe, "Estimator models for combining vector estimators," *Math. Comput. Model.*, vol. 12, no. 12, pp. 1627–1642, 1989.

[28] F. Lavancier and P. Rochet, "A general procedure to combine estimators," *Comput. Statist. Data Anal.*, vol. 94, pp. 175–192, Feb. 2016.

[29] T. Blu and F. Luisier, "The SURE-LET approach to image denoising," *IEEE Trans. Image Process.*, vol. 16, no. 11, pp. 2778–2786, Nov. 2007.

[30] K. N. Chaudhury and K. Rithwik, "Image denoising using optimally weighted bilateral filters: A sure and fast approach," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sept. 2015, pp. 108–112.

[31] J. Jancsary, S. Nowozin, and C. Rother, "Loss-specific training of non-parametric image restoration models: A new state of the art," in *Proc. 12th Eur Conf. Comput. Vis. (ECCV)*, Oct. 2012, pp. 112–125.

[32] F. Agostinelli, M. R. Anderson, and H. Lee, "Adaptive multi-column deep neural networks with application to robust image denoising," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2013, pp. 1493–1501.

[33] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep CNN denoiser prior for image restoration," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2808–2817.

[34] S. H. Chan, X. Wang, and O. A. Elgendy, "Plug-and-Play ADMM for image restoration: Fixed-point convergence and applications," *IEEE Trans. Comput. Imag.*, vol. 3, no. 1, pp. 84–98, Mar. 2017.

[35] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[36] M. Grant and S. P. Boyd. (Mar. 2014). *CVX: MATLAB Software for Disciplined Convex Programming, Version 2.1*. [Online]. Available: http://cvxr.com/cvx

[37] M. Grant and S. P. Boyd, "Graph implementations for nonsmooth convex programs," in *Proc. Recent Adv. Learn. Control*. London, U.K.: Springer, 2008, pp. 95–110.

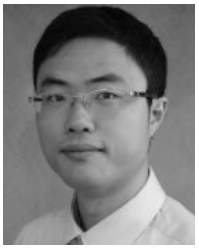
[38] L. Condat, "Least-squares on the simplex for multispectral unmixing," Res. Rep, GIPSA-Lab, Univ. Grenoble Alpes, Grenoble, France, Feb. 2017. Accessed: Mar. 20, 2019. [Online]. Available: http://www.gipsa-lab.fr/~laurent.condat/publis/Condat-unmixing.pdf

[39] J. Mairal, "Optimization with first-order surrogate functions," in *Proc. 30th Int. Conf. Mach. Learn. (ICML)*, 2013, pp. 783–791.

[40] M. Jaggi. (Dec. 2011). "Convex optimization without projection steps." [Online]. Available: https://arxiv.org/abs/1108.1170
[41] C.-A. Deledalle, L. Denis, S. Tabti, and F. Tupin, "Closed-form expressions of the eigen decomposition of 2×2 and 3×3 Hermitian matrices," Univ. de Lyon, Lyon, France, Res. Rep, 2017. Accessed: Mar. 20, 2019. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01501221
[42] F. Luisier, T. Blu, and M. Unser, "A new SURE approach to image denoising: Interscale orthonormal wavelet thresholding," *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 593–606, Mar. 2007.
[43] S. Ramani, T. Blu, and M. Unser, "Monte-carlo Sure: A black-box optimization of regularization parameters for general denoising algorithms," *IEEE Trans. Image Process.*, vol. 17, no. 9, pp. 1540–1554, Sep. 2008.
[44] S. Cha and T. Moon, "Neural adaptive image denoiser," in *Proc. IEEE Int Conf. Acoust. Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 2981–2985.
[45] A. Foi, "Clipped noisy images: Heteroskedastic modeling and practical denoising," *Signal Process.*, vol. 89, no. 12, pp. 2609–2629, 2009.
[46] L. Kang, P. Ye, Y. Li, and D. Doermann, "Convolutional neural networks for no-reference image quality assessment," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, pp. 1733–1740.
[47] Y. Li, L. M. Po, L. Feng, and F. Yuan, "No-reference image quality assessment with deep convolutional neural networks," in *Proc. IEEE Int. Conf. Digit. Signal Process. (DSP)*, Oct. 2016, pp. 685–689.
[48] S. Bosse, D. Maniry, T. Wiegand, and W. Samek, "A deep neural network for image quality assessment," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sept. 2016, pp. 3773–3777.
[49] J. Kim, H. Zeng, D. Ghadiyaram, S. Lee, L. Zhang, and A. C. Bovik, "Deep convolutional neural models for picture-quality prediction: Challenges and solutions to data-driven image quality assessment," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 130–141, Nov. 2017.
[50] Y. Li, X. Ye, and Y. Li, "Image quality assessment using deep convolutional networks," *AIP Adv.*, vol. 7, no. 12, Dec. 2017. Art. no. 125324.
[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, Dec. 2014. Accessed: Mar. 20, 2019. [Online]. Available: https://arxiv.org/abs/1412.6980
[52] J. W. Tukey, *Exploratory Data Analysis*, vol. 2. Reading, MA, USA: Addison-Wesley, 1977.
[53] P. Bühlmann, and B. Yu, "Boosting with the l_2 loss," *J. Amer. Statist. Assoc.*, vol. 98, no. 462, pp. 324–339, 2003.
[54] M. R. Charest and P. Milanfar, "On iterative regularization and its application," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 3, pp. 406–411, Mar. 2008.
[55] H. Talebi, X. Zhu, and P. Milanfar, "How to SAIF-ly boost denoising performance," *IEEE Trans. Image Process.*, vol. 22, no. 4, pp. 1470–1485, Apr. 2013.
[56] M. R. Charest, M. Elad, and P. Milanfar, "A general iterative regularization framework for image denoising," in *Proc. 40th Annu. Conf. Inf. Sci. Syst. (CISS)*, Mar. 2006, pp. 452–457.
[57] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin, "An iterative regularization method for total variation-based image restoration," *Multiscale Model Simul.*, vol. 4, no. 2, pp. 460–489, 2005.
[58] Y. Romano and M. Elad, "Boosting of image denoising algorithms," *SIAM J. Imag. Sci.*, vol. 8, no. 2, pp. 1187–1219, 2015.
[59] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.
[60] L. Azzari and A. Foi, "Gaussian-cauchy mixture modeling for robust signal-dependent noise estimation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2014, pp. 5357–5361.
[61] J. Zhang and K. Hirakawa, "Improved denoising via Poisson mixture modeling of image sensor noise," *IEEE Trans. Image Process.*, vol. 26, no. 4, pp. 1565–1578, Apr. 2017.
[62] W. Cheng and K. Hirakawa, "Towards optimal denoising of image contrast," *IEEE Trans. Image Process.*, vol. 27, no. 7, pp. 3446–3458, Jul. 2018.
[63] J. Xu, L. Zhang, D. Zhang, and X. Feng, "Multi-channel weighted nuclear norm minimization for real color image denoising," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1096–1104.
[64] J. Xu, L. Zhang, and D. Zhang, "External prior guided internal prior learning for real-world noisy image denoising," *IEEE Trans. Image Process.*, vol. 27, no. 6, pp. 2996–3010, Jun. 2018.
[65] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Trans. Comput. Imag.*, vol. 3, no. 1, pp. 47–57, Mar. 2017.
[66] E. Luo, S. H. Chan, and T. Q. Nguyen, "Adaptive image denoising by targeted databases," *IEEE Trans. Image Process.*, vol. 24, no. 7, pp. 2167–2181, Jul. 2015.
[67] E. Luo, S. H. Chan, and T. Q. Nguyen, "Adaptive image denoising by mixture adaptation," *IEEE Trans. Image Process.*, vol. 25, no. 10, pp. 4489–4503, Oct. 2016.
[68] S. Boyd and L. El Ghaoui, "Method of centers for minimizing generalized eigenvalues," *Linear Algebra Appl*, vols. 188–189, pp. 63–111, Jul./Aug. 1993.

Joon Hee Choi (S'18–M'19) received the B.E. degree in electronics engineering and the B.B.A. degree in business administration from Sogang University, South Korea, in 2004, and the M.S. degree and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2014 and 2018, respectively. From 2003 to 2012, he was an Assistant Manager with Samsung Electronics, Co., Ltd. He is currently a Staff Engineer with Qualcomm Inc. His research interests include machine learning, image/video restoration, computer vision, and optimization algorithms for large-scale datasets.

Omar A. Elgendy (S'15) received the B.Sc. degree (Hons.) in electronics and communications engineering and the M.Sc. degree in engineering mathematics from the Faculty of Engineering, Cairo University, in 2010 and 2015, respectively. He is currently pursuing the Ph.D. degree with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN. His research interests include statistical image processing with applications to image reconstruction using single-photon imaging sensors, statistical pattern classification, deep learning, and large-scale optimization. He received the Best Paper Award from the 2016 IEEE International Conference on Image Processing and the Bilsland Dissertation Fellowship for the academic year 2018-2019 from Purdue University.



Stanley H. Chan (S'06–M'12–SM'17) received the B.Eng. degree in electrical engineering from The University of Hong Kong in 2007 and the M.A. degree in mathematics and the Ph.D. degree in electrical engineering from the University of California at San Diego, La Jolla, CA, in 2009 and 2011, respectively.

From 2012 to 2014, he was a Post-Doctoral Research Fellow with Harvard University, Cambridge, MA. In 2014, he joined Purdue University, West Lafayette, IN, where he is currently an Assistant Professor of electrical and computer engineering and an Assistant Professor of statistics. His research interests include signal and image processing, applied statistics, and large-scale numerical optimization.

Dr. Chan has been an elected member of the IEEE Signal Processing Society Computational Imaging Technical Committee since 2015. He was a recipient of the Best Paper Award of the 2016 IEEE International Conference on Image Processing for his work on single-photon image sensors. He was also a recipient of multiple education awards, including the Eta Kappa Nu Teaching Award in 2015, the 2016 IEEE Signal Processing Cup Second Prize, the Purdue College of Engineering Outstanding Graduate Mentor Award in 2016, the Purdue Teaching for Tomorrow Fellowship in 2018, the Eta Kappa Nu Outstanding Professor Award in 2018, and the Purdue College of Engineering Exceptional Early Career Teaching Award in 2019. He was also a recipient of the Croucher Foundation Scholarship for Ph.D. Studies (2008–2010) and the Croucher Foundation Fellowship for Postdoctoral Research (2012–2013). He was the Co-Chair and the Co-Organizer of the computational imaging special session of ICIP 2016. He has served on multiple technical program committees, including ICIP, ICASSP, OSA Imaging and Applied Optics Congress, and Midwest Machine Learning Symposium. He was an Associate Editor of *Optics Express* (OSA) from 2016 to 2018. He has been an Associate Editor of the IEEE TRANSACTIONS ON COMPUTATIONAL IMAGING since 2018.