Hyperspectral Image Denoising Using SURE-Based Unsupervised Convolutional Neural Networks

Han V. Nguyen, *Graduate Student Member, IEEE*, Magnus O. Ulfarsson, *Senior Member, IEEE*, and Johannes R. Sveinsson, *Senior Member, IEEE*

***Abstract*— Hyperspectral images (HSIs) are useful for many remote sensing applications. However, they are usually affected by noise that degrades the HSIs quality. Therefore, HSI denoising is important to improve the performance of subsequent HSI processing and analysis. In this article, we propose an HSI denoising method called Stein's unbiased risk estimate-convolutional neural network (SURE-CNN). The method is based on an unsupervised CNN and SURE. The main difference of SURE-CNN from existing supervised learning methods is that the SURE-based loss function can be computed only from noisy data. Since SURE is an unbiased estimate of the mean squared error (MSE) of an estimator, training a CNN using the SURE loss can yield similar results as using the MSE with ground truth in supervised learning. Also, a subspace version of SURE-CNN is proposed to reduce the running time. Extensive experimental results with both simulated and real data sets show that the SURE-CNN method outperforms the competitive methods in both objective and subjective assessments.**

***Index Terms*— Convolutional neural networks (CNNs), hyperspectral (HSI) image denoising, Stein's unbiased risk estimate (SURE), unsupervised deep learning (DL).**

I. Introduction

HYPERSPECTRAL images (HSIs) acquired by an optical remote sensing system provide both spatial and spectral information of a scene. It is distinct from other kinds of remote sensing modalities due to the rich spectral information given by hundreds of contiguous bands. The spectral signals observed by an HSI are useful for discriminating between the materials in the scene. Therefore, HSIs are useful in diverse applications, such as environmental monitoring, mapping, mineral mining, target detection and classification, and many more [1]–[3].

An HSI has high spectral resolution and low signal-to-noise-ratio (SNR) within a band. The remote sensing HSI sensors are usually mounted in an airborne craft or a satellite and work in a harsh environment. As a result, an HSI is easily contaminated by noise, which is mainly caused by atmospheric absorption, photon effects, and instrument malfunctions. The main types of noise are the Gaussian noise, Poissonian noise, stripped noise, and missing pixels noise [4]. Therefore, denoising is an important process to improve the quality of the HSI applications.

Several HSI denoising methods exist. The most straightforward techniques apply classical 2-D image denoising method band-by-band. Examples of those methods are the block-matching and 3-D filtering (BM3D) algorithm [5], the weighted nuclear norm minimization (WNNM) algorithm [6], and a learning model such as expected patch log-likelihood (EPLL) [7]. The bandwise methods are simple, but since they do not utilize spectral information, they usually lead to a sizeable spectral distortion.

The high correlation in spatial and spectral domains motivates multiband denoising methods. The first category of those methods [8]–[11] exploits the sparsity in a transform domain, such as the principal component analysis (PCA), Fourier, and wavelet domains to separate clean images from noise. A majority of multiband HSI denoising methods are model-based. In these methods, the denoising problem is formulated as an inverse problem and solved by using penalized regression. The solution is to minimize a cost function, which is the sum of a fidelity term and one or more regularization terms. Generally, prior knowledge about spatial and spectral correlations determines the regularization terms and denoising methods. The spectral–spatial total variation (TV) was used in [12]–[15] to deal with the Gaussian and sparse noises. Lu *et al.* [16], Li *et al.* [17], and Bai *et al.* [18] proposed denoising methods based on sparse representation, which utilizes the nonlocal self-similarity in a band and between bands. Another idea exploiting the redundancy of HSI data is to employ a low-rank model. A low-rank approximation using the Tucker3 tensor decomposition was proposed in [19] for both HSI denoising and dimensionality reduction. A similar low-rank technique based on using learning tensor decomposition approach was proposed in [20]. Zhang *et al.* [21] presented an HSI restoration technique based on the low-rank matrix recovery (LRMR). An extension of the LRMR algorithm [22] using an iterative noise adjusted procedure worked well for removing mixed types of noise in HSI. More recently, the combination of low-rank and sparse representation proposed in [23]–[26] yielded state-of-the-art results. Although multiband methods often perform well, it is hard to select the model parameters and regularizations since they are data-dependent.

In the last decade, many HSI denoising methods based on deep learning (DL) have been proposed. They are inspired

Manuscript received March 24, 2020; revised June 1, 2020; accepted July 7, 2020. Date of publication July 23, 2020; date of current version March 25, 2021. This work was supported in part by the Icelandic Research Fund under Grant 174075-05 and Grant 207233-051, and in part by the University of Iceland Doctoral Fund under Grant 1547-154305. *(Corresponding author: Magnus O. Ulfarsson.)*

The authors are with the Faculty of Electrical and Computer Engineering, University of Iceland, 107 Reykjavik, Iceland (e-mail: hvn2@hi.is; mou@hi.is).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2020.3008844

by the DL-based denoisers, which were developed for 2-D images [27]–[29]. Most of those methods are supervised, i.e., they are trained by using many noisy-clean HSI pairs. For example, Yuan *et al.* [30] proposed an HSI denoising method employing a spectral–spatial deep residual convolutional neural network (HSID-CNN). Maffei *et al.* [31] proposed a similar denoising method using a single CNN (HSI-SDeCNN). Both HSID-CNN and HSI-SDeCNN are trained by using noisy-clean patches extracted from a remote sensing HSI data set. However, in remote sensing, it is costly to acquire a clean HSI. Also, training a deep CNN with limited data may not generalize well. To overcome this problem, the methods [32]–[34] train the CNNs using natural HSIs, which are taken by a hyperspectral camera. After that, they are transferred and fine-tuned to work with remote sensing HSIs. Beyond the Gaussian noise removal, a denoising methods based on DL for non-i.i.d. Gaussian and hybrid noise have been proposed in [35]. Few unsupervised learning HSI denoising methods can be found in the academic literature. The representatives are the deep hyperspectral prior method [36] and the zero-shot learning approach [37].

In this article, we propose an unsupervised DL-based HSI denoising method that utilizes Stein's unbiased risk estimate (SURE) [38], [39]. Instead of using the mean-squared-error (MSE), which requires ground truth, as a loss function, the proposed network uses SURE. The advantage is that SURE is an unbiased estimate of the MSE and can be computed only by using the noisy HSI. The main contributions of this article are summarized as follows.

1) We propose an unsupervised CNN using a SURE loss function. Since the SURE loss function can prevent overfitting, SURE-CNN can be trained and tested at the same time without validation.
2) We develop a SURE loss function for HSI taking into account band-varying noise. The HSI SURE loss function is verified to work with unknown noise variance and different CNN architectures.
3) We incorporate a dimensionality reduction step using SVD into the SURE-CNN denoising process. This step can significantly reduce the running time and may improve denoising performance.

The rest of this article is organized as follows. Section II contains the HSI denoising problem formulation. We propose the SURE-CNN denoising method in Section III. Section IV shows the experimental results. Finally, Section V concludes this article.

II. HSI Denoising Problem Formulation

A. HSI Denoising Model

An HSI in its original form is 3-D volumetric data with two spatial and one spectral dimension. A clean HSI and its noisy observation are denoted in matrix forms as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_B] \in \mathbb{R}^{n\times B}$ and $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_B] \in \mathbb{R}^{n\times B}$, respectively. Here, $\mathbf{x}_i, \mathbf{y}_i \in \mathbb{R}^n$ are the ith vectorized clean and noisy observation bands, respectively, n is the number of pixels, and B is the number of bands. We consider a noisy additive model, where the noisy observation and the (unknown) clean HSI are related as

$$\mathbf{Y} = \mathbf{X} + \mathbf{N} \tag{1}$$

where $\mathbf{N} = [\mathbf{n}_1, \mathbf{n}_2, \ldots, \mathbf{n}_B] \in \mathbb{R}^{n\times B}$ is the additive noise. The $\mathbf{n}_i$ noise vector is assumed to be drawn from a zero mean, isotropic Gaussian distribution, i.e., $\mathbf{n}_i \sim \mathcal{N}(0, \sigma_i^2\mathbf{I}_n)$. Therefore, the HSI denoising problem is to find an estimate $\widehat{\mathbf{X}} \in \mathbb{R}^{n\times B}$ of the clean HSI $\mathbf{X}$ from its noisy observation $\mathbf{Y}$.

B. Related Works

DL is an active research topic in HSI processing and analysis for remote sensing applications [40], [41]. Due to the lack of clean HSIs, the use of supervised learning is limited. On the other hand, the unsupervised learning methods do not require any clean HSIs. An example is the zero-shot HSI denoising using a 3-D separable CNN [37]. In this method, synthetic noise is added to an HSI to generate another noisy HSI. Many HSI and noisy HSI patches are used to train a CNN. After training, the CNN is used for denoising the original noisy HSI. The theoretical foundation behind zero-shot HSI denoising method is from the Noise2Noise [42] and Noise2Void [43]. Noise2Noise and Noise2Void show that training a CNN using independent pairs of noisy images gives the same result as using the noisy-clean image pairs.

Another example of an unsupervised learning HSI denoising technique is the deep hyperspectral prior (DIP-HSI) [36], which can be used for HSI denoising, inpainting, and super-resolution. DIP-HSI is an extension of the deep image prior (DIP) method [44]. It was concluded from [44] that the structure of a deep CNN yields good prior to RGB image denoising, inpainting, and super-resolution. This is because CNN offers high impedance to noise and low impedance to an image. As a result, during the optimization process, the solution fits the image before the noise. However, DIP is prone to overfitting since the optimal training point is hard to choose. One way of alleviating this problem is by explicitly introducing the penalties, such as the TV [45] and regularization by denoising [46]. Another way is to train the neural network by minimizing the MSE that is indirectly computed using SURE [47]–[49]. The results show that training a neural network with SURE yields similar performance as with the MSE and ground truth.

The proposed SURE-CNN for HSI denoising method is an extension of the idea of [47] and [48], which focuses on RGB images. The distinction of our proposed method is the derivation of the SURE loss for HSI and that we train and test a CNN based on SURE for a single HSI at the same time rather than training a generalized network using a large data set as in [48].

III. Proposed SURE-Based CNN for HSI Denoising

Given a noisy HSI observation as in (1), we propose an unsupervised CNN-based denoiser $f_\Theta(\cdot)$ that takes the noisy observation as the input and directly estimate the output, i.e., $\widehat{\mathbf{X}} = f_\Theta(\mathbf{Y})$, where Θ is the network parameter. The key idea of the proposed method is to train the CNN using a SURE loss function.

A. SURE for HSI Denoising

Assume that a vectorized image $\mathbf{y}_i \in \mathbb{R}^n, i = 1, \ldots, B$ at the ith band is drawn from (1). The estimated $\widehat{\mathbf{x}}_i$ from $\mathbf{y}_i$ is the ith vectorized output of SURE-CNN, i.e., $\widehat{\mathbf{x}}_i = f_\Theta(\mathbf{Y})_i$. SURE $\widehat{R}$ of the estimator $f_\Theta(\cdot)$ is given by

$$\widehat{R} = \frac{1}{B}\sum_{i=1}^{B}\left(\frac{1}{n}\|\mathbf{y}_i - f_\Theta(\mathbf{Y})_i\|_2^2 - \sigma_i^2 + \frac{2\sigma_i^2}{n}\mathrm{tr}\left(\frac{\partial f_\Theta(\mathbf{Y})_i}{\partial \mathbf{y}_i^T}\right)\right). \tag{2}$$

SURE is an unbiased estimate of the risk (true MSE) $R = (1/nB)\sum_{i=1}^{B} E[\|\mathbf{x}_i - f_\Theta(\mathbf{Y})_i\|^2]$, where $E[\cdot]$ is the expectation with respect to noise. In practice, we would like to train the network using the true MSE as loss; however, R is not computable since $\mathbf{x}_i$ is unknown. Therefore, we use the SURE loss instead.

Unfortunately, the explicit computation of the last term of (2) is only possible in special cases, such as linear and coordinatewise nonlinear estimators. For any arbitrary nonlinear estimator, such as a neural network estimator, using the Monte Carlo SURE introduced in [50], the trace term in (2) can be approximated as

$$\mathrm{tr}\left(\frac{\partial f_\Theta(\mathbf{Y})_i}{\partial \mathbf{y}_i^T}\right) \approx \mathbf{b}^T\left(\frac{f_\Theta(\mathbf{Y}+\epsilon\mathbf{b})_i - f_\Theta(\mathbf{Y})_i}{\epsilon}\right) \tag{3}$$

where $\mathbf{b}$ is an i.i.d. Gaussian distribution with zero mean and unit variance, and ϵ is a small value that can be chosen in the range of 10^{-5}–10^{-2}.

The noise standard deviation of each band, σ_i, can be estimated using the HySime algorithm [51] or the median absolute deviation estimator in the highest subband (HH) of the wavelet transform of each band as in [52]

$$\widehat{\sigma}_i = \frac{\mathrm{median}\left(\left|\mathbf{W}_{(i)}^{\mathrm{HH}}\right|\right)}{0.6745}, \quad i = 1, \ldots, B. \tag{4}$$

In the experiments, we also treat a special case where the noise variance is isotropic, i.e., $\sigma_i^2 = \sigma^2$. In that case, we estimate the noise standard deviation σ by taking the average over the estimates in (4).

Theoretically, the assumption under SURE is the Gaussian distribution. However, we can extend SURE-CNN to deal with non-Gaussian noise, such as the Poissonian noise [25]. For the Poissonian noise, we apply the Anscombe transform [53] to convert it to approximate i.i.d. Gaussian noise. Then, we apply SURE-CNN to denoise the i.i.d. Gaussian and inverse-Anscombe transforms to obtain the denoised HSI.

Finally, we formulate the SURE loss function to train the unsupervised CNN is (2) with the trace term as in (3) and the estimated noise standard deviation as in (4).

B. Proposed CNN Architecture

In our preliminary work [54], we used a CNN with skip (shortcut) connections for denoising of an HSI that was perturbed by the isotropic Gaussian noise. The proposed architecture was inspired by the skip connection CNNs that were used in image restoration, segmentation, and recognition. The skip connection architecture helps learning and makes convergence faster [55]–[58]. Here, we use the skip connection CNN with an extensive study on the network depth and the skip connection layers to get the best network structure and parameters for the HSI denoising problem.

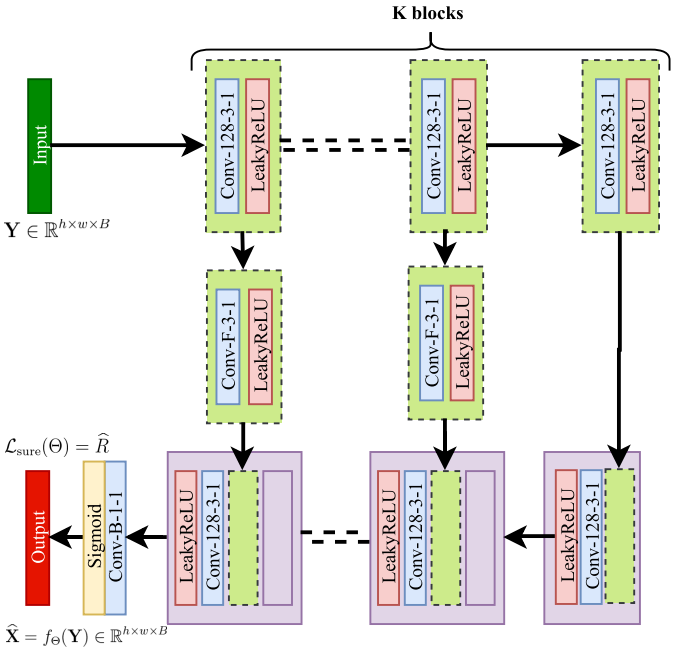


Fig. 1. Network structure. Conv-128-3-1 represents a convolutional layer with 128 filters of Kernel size 3 and stride 1. The number of conv-relu blocks is $K = 5$, and the number of filters in a skip connection layer is $F = 5$.

Fig. 1 shows the proposed network structure. It consists of three main parts: the input branch, the output branch, and the skip connection. The input and output branches are symmetrically designed, whose K blocks of convolutional and LeakyReLU (conv-relu) layers are stacked consecutively. The convolutional layers in the input and output branches have 128 filters, a filter size of 3, and a stride of 1. The LeakyReLU layer acts as an activation function. The last convolutional layer in the output branch uses B filters, the filter size and strides of 1, and the Sigmoid activation function. The downsampling and upsampling layers can capture multiscaled features and accelerate the computation, but they may lose information. Thus, downsampling and upsampling layers are not used in the design. Instead, the spatial size of the HSI is kept constant throughout the network. A skip connection is a convolutional layer with F filters; the filter size, stride, and activation function are the same as the convolutional layer in the input and output branches. It connects a conv-relu block in the input branch to another conv-relu block in the output branch. We train the CNN in an unsupervised manner using the SURE loss as in (2), and a noisy HSI is used as both network input and output. An Adam optimizer, with a learning rate of 0.001, is used, and the network is implemented using Tensorflow 2.0.

To find a network depth which best suits to the HSI denoising problem, we assess the effects of the numbers of conv-relu blocks K and the numbers of filters F in the skip connection layers to the peak-SNR (PSNR) during training. We use a small part of the simulated Pavia University (PU)

TABLE I

SURE-CNN PERFORMANCE GIVEN BY PSNR IN dB FOR DIFFERENT CONVOLUTIONAL LAYER BLOCKS K AND NUMBER OF FILTERS F IN THE SKIP CONNECTION LAYERS FOR VARIOUS CASES OF NOISE. THE RESULTS ARE THE AVERAGE VALUES OVER TEN RUNS

PSNR (dB)		Case 1: $\sigma = 0.3$	Case 2: $\sigma = 1$, $\eta = 20$	Case 3: $\sigma \sim \mathcal{U}(0.1, 0.2)$
$F = 5$	$K = 3$	30.90	37.23	34.94
	$K = 5$	31.45	37.34	35.29
	$K = 7$	31.50	37.03	35.37
$K = 5$	$F = 5$	31.45	37.34	35.29
	$F = 25$	30.90	36.89	35.05
	$F = 64$	30.67	36.93	34.78

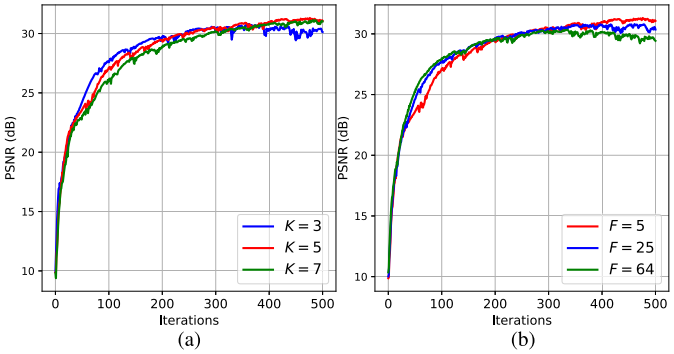


Fig. 2. PSNR as a function of iterations by SURE-CNN for different numbers conv-relu blocks K and number of filters F in the skip connection layers. The results are the average values over ten runs. (a) $K \in \{3,\ 5,\ 7\}$ and $F = 5$. (b) $F \in \{\ 5,\ 25,\ 64\}$ and $K = 5$.

data set with different noise scenarios for the experiments (refer to Section IV-A for the PSNR definition and data description). Fig. 2(a) and (b) shows the PSNR as a function of training iterations for $K \in \{3,\ 5,\ 7\}$ and $F = 5$ and for $F \in \{5,\ 25,\ 64\}$ and $K = 5$, respectively. Table I gives the PSNR for the networks using various values of K and F for different cases of noise. By addressing the network depth and the number of filters in the skip connection layers, the network with five conv-relu blocks ($K = 5$) and five filters ($F = 5$) in the skip connection is chosen since it gives the best tradeoff between PSNR and the network complexity.

C. Training and Testing

SURE-CNN is unsupervised. It allows us to train and test the network at the same time. The input of SURE-CNN is a 3-D noisy HSI, and the output is a 3-D denoised HSI. Validation is not needed since the SURE loss is an unbiased estimate of the true MSE. After convergence, we stop training and use the pretrained network to predict the denoised HSI. We use an early stopping schedule. If the training loss does not decrease after a predefined iteration, the training will stop.

D. Subspace SURE-CNN for HSI Denoising

The SURE-CNN denoising method works with big volumetric data. By utilizing the spectral redundancy of HSI data, we perform dimensionality reduction on the noisy observation data to reduce computational time. Recalling that $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_B]$, we can write (by using SVD)

$$\mathbf{Y} \approx \mathbf{Z}\mathbf{E}^T$$

where $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k] \in \mathbb{R}^{B\times k} (k \ll B)$, and $\mathbf{E}^T\mathbf{E} = \mathbf{I}_k$. The eigenimages $\mathbf{Z}$ can be computed as $\mathbf{Z} \approx \mathbf{Y}\mathbf{E}$. In the additive Gaussian noise case, the eigenimages follow the additive Gaussian model (1), as the original data [25]. Thus, the SURE-CNN denoising method can be applied to the eigenimages. Finally, the the denoised eigenimages are transformed back to the original HSI space. The subspace SURE-CNN HSI denoising is described in Algorithm 1.

Algorithm 1 Subspace SURE-CNN HSI Denoising Algorithm

1) **Input:** Noisy HSI $\mathbf{Y}$, dimension of subspace k.
2) Find a subspace base $\mathbf{E}$ using SVD, compute noisy eigenimages $\mathbf{Z} = \mathbf{Y}\mathbf{E}$.
3) Denoise noisy eigenimages using SURE-CNN, $\widehat{\mathbf{Z}} = f_\theta(\mathbf{Z})$.
4) Transform denoised eigenimages to denoised HSI $\widehat{\mathbf{X}} = \widehat{\mathbf{Z}}\mathbf{E}^T$.

IV. EXPERIMENTAL RESULTS

A. Data Description

In the following experiments, we use two simulated and two real HSI data sets. The first simulated data set is the Washington DC Mall (DC) data set collected by the Hyperspectral Digital Imagery Collection Experiment (HYDICE). It has 1208×307 pixels and 191 bands. The second simulated data set is the Pavia University (PU) data set collected by the Reflective Optics System Imaging Spectrometer (ROSIS-03). It has 103 bands of 640×340 pixels. In the experiments, we use small parts of the DC and PU data sets. Their sizes are $400 \times 200 \times 191$ for the DC data set and $400 \times 200 \times 103$ for the PU data set. The DC and PU data sets are assumed to be noise-free HSIs. They are normalized band-by-band between 0 and 1 as in [21] and [25] before adding the noise. The noisy simulated data sets are created as follows.

1) *Case 1:* Isotropic Gaussian noise where $\sigma_i^2 = \sigma^2$ with zero mean and standard deviation σ is added to each band. We consider the following values of standard deviation, $\sigma \in \{0.05,\ 0.1,\ 0.2,\ 0.3\}$.
2) *Case 2:* Bandwise Gaussian noise with zero mean and variance that varies according to a bell shape is added to each band. The variance varies according to

$$\sigma_i^2 = \sigma^2 \frac{e^{-\frac{(i-B/2)^2}{2\eta^2}}}{\sum_{j=1}^{B} e^{-\frac{(i-B/2)^2}{2\eta^2}}} \tag{5}$$

where $\sigma = 1$ and $\eta = 20$ are two numbers that control the noise intensity and the bell width, respectively.
3) *Case 3:* A zero mean Gaussian noise is added to each band. The standard deviation for each band is drawn from a uniform distribution between 0.1 and 0.2 [$\sigma \sim \mathcal{U}(0.1, 0.2)$].

The two real data are the Indian Pines (IP) and the Urban (UB) data sets. The IP data set is a $145 \times 145 \times 220$ HSI obtained by the airborne visible/infrared imaging spectrometer (AVIRIS). The UB data set is a $307 \times 307 \times 210$ HSI obtained by the HYDICE. Both IP and UB are contaminated by various types of noise, such as the Gaussian, Poissonian, stripped, and missing pixels noises.

The following evaluated metrics are used. First, the PSNR given by

$$\text{PSNR} = 10\log_{10}\left(\frac{\max^2(\mathbf{X})}{\text{MSE}(\mathbf{X}, \widehat{\mathbf{X}})}\right)$$

where $\text{MSE}(\mathbf{X}, \widehat{\mathbf{X}})$ is the MSE between $\mathbf{X}$ and $\widehat{\mathbf{X}}$. Second, the mean structural similarity index (MSSIM) is the bandwise mean of SSIM. The SSIM of ith band is calculated as [59]

$$\text{SSIM}_{i,j} = \frac{(2\mu_{x_{i,j}}\mu_{\widehat{x}_{i,j}} + c_1)(2\sigma_{x_{i,j}\widehat{x}_{i,j}} + c_2)}{\left(\mu^2_{x_{i,j}} + \mu^2_{\widehat{x}_{i,j}} + c_1\right)\left(\sigma^2_{x_{i,j}} + \sigma^2_{\widehat{x}_{i,j}} + c_2\right)}$$

where $\mu_{x_{i,j}}, \mu_{\widehat{x}_{i,j}}, \sigma_{x_{i,j}}$, and $\sigma_{\widehat{x}_{i,j}}$ denotes mean and standard deviation for the reference and estimated images in a local window centered at pixel j, respectively. $\sigma_{x_{i,j}\widehat{x}_{i,j}}$ is the cross-covariance between two images. $c_1 = (K_1 D)^2$ and $c_2 = (K_2 D)^2$, where $K_1 = 0.01, K_2 = 0.3$, and D is the dynamical range of the image that is set to 1 in this article. Finally, the spectral angle mapper (SAM) in degrees

$$\text{SAM} = \frac{1}{nB}\sum_{p=1}^{nB} \arccos\left(\frac{\mathbf{x}^T_{(p)}\widehat{\mathbf{x}}_{(p)}}{\|\mathbf{x}_{(p)}\|\|\widehat{\mathbf{x}}_{(p)}\|}\right)\frac{180}{\pi}$$

where $\mathbf{x}_{(p)}$ and $\widehat{\mathbf{x}}_{(p)}$ are the reference image and estimated image at pixel p, respectively.

B. Parameters Validation

In this section, we verify the performance of SURE as an approximator of the true MSE. Fig. 3 shows the training evolution of the skip connection CNN ($K = 5, F = 5$) using the SURE loss and the fidelity loss, i.e., $\mathcal{L}_{\text{fide}}(\Theta) = (1/nB)\|\mathbf{Y} - f_\Theta(\mathbf{Y})\|_2^2$, for the DC and PU data sets in denoising Case 1 with $\sigma = 0.3$. It can be seen that the SURE loss approximates almost perfectly the true MSE, while the fidelity loss fails to do so. Furthermore, the network that uses fidelity loss tends to overfit since after 300 iterations (for PU) and 400 iterations (for DC), the training loss decreases while the true MSE starts increasing. Hence, training the network with fidelity loss needs a validation scheme, such as early stopping, to reach an optimal point. On the other hand, SURE loss does not require the validation scheme.

Next, we assess the effects on the SURE loss by estimating $\widehat{\sigma}_i$ with wavelet (4) and the HySime algorithm. Fig. 4(a) and (c) shows the true σ_i in (5) and their estimated values by wavelet and HySime for the PU and DC data sets, respectively. Fig. 4(b) and (d) shows the SURE loss using three abovementioned σ_i in a skip connection CNN ($K = 5$ and $F = 5$), for denoising Case 2 of the PU and DC data sets, respectively. It is observed from these figures that both wavelet and HySime give a relatively good estimate of $\widehat{\sigma}_i$, and there is almost no difference in the SURE loss between using true σ_i and estimated $\widehat{\sigma}_i$.

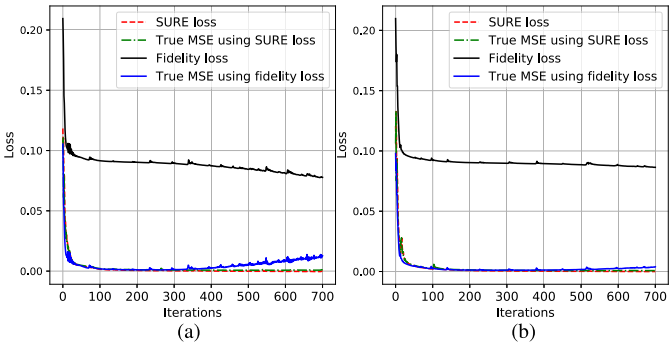


Fig. 3. Training loss of the skip connection CNN using both SURE loss and fidelity loss for denoising Case 1 with $\sigma = 0.3$. (a) PU data set. (b) DC data set.

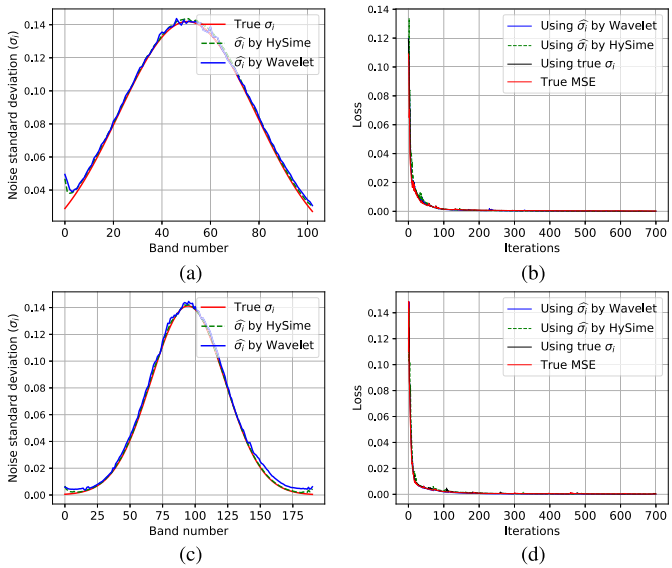


Fig. 4. Influence of the estimated noise standard deviation on the training SURE loss for denoising Case 2 with $\sigma = 1, \eta = 20$. (a) Estimated noise standard deviation for the PU data set. (b) Training loss using the true and estimated noise standard deviation for the PU data set. (c) Estimated noise standard deviation for the DC data set. (d) Training loss using the true and estimated noise standard deviation for the DC data set.

We further evaluate the SURE loss for different CNN architectures. Fig. 5 shows the SURE loss and the true MSE during training using the modified UNet [44] and the ResNet [60] for denoising of the PU and DC data sets in Case 1 with $\sigma = 0.3$. Here, we change the input and output channels of the UNet and the ResNet to fit HSI data. For the UNet, the number of filters in the skip connections is four. The number of residual blocks in the ResNet is five. It is clearly verified that the SURE loss works well for both UNet and ResNet structures. The PSNR, for each data set and network, shows that the UNet gives approximately 2 dB higher than the ResNet for the same number of training iterations.

C. Performance: Simulated Data Sets

Here, we evaluate the performance of SURE-CNN for both quantitative metrics and visualization of the denoised images. SURE-CNN is compared with the following competitive methods: two HSI denoising methods based on tensor decomposition that are a low-rank tensor approximation

TABLE II

DENOISING FOR THE DC DATA SET USING DIFFERENT METHODS. THE EVALUATED METRICS ARE PSNR IN dB, AND MSSIM AND SAM IN DEGREES. THE RESULTS ARE THE AVERAGE VALUES OVER TEN RUNS. THE STANDARD DEVIATIONS FOR PSNR, MSSIM, AND SAM IN EACH METHOD ARE LESS THAN 0.1 dB, 0.0008, AND 0.03°, RESPECTIVELY. BEST RESULTS ARE IN BOLDFACE

DC	Noise level	Metric	Noisy	LRTA	TDL	FORPDN	BM4D	NAILRMA	HyRes	FastHyDe	SURE-CNN
Case 1	$\sigma = 0.05$	PSNR	26.02	37.65	40.43	35.27	36.54	40.73	40.49	**41.73**	41.32
		MSSIM	0.628	0.962	0.981	0.946	0.955	0.977	0.979	0.983	**0.987**
		SAM	10.910	2.626	1.781	2.919	3.516	1.765	1.744	**1.506**	1.555
	$\sigma = 0.1$	PSNR	20.00	34.13	35.24	32.39	32.49	36.20	36.62	37.75	**37.89**
		MSSIM	0.3617	0.934	0.952	0.898	0.899	0.942	0.949	0.962	**0.973**
		SAM	20.977	3.595	3.005	4.521	4.881	2.816	2.645	**2.266**	**2.266**
	$\sigma = 0.2$	PSNR	13.98	30.18	30.41	29.39	28.78	31.23	32.15	33.98	**34.15**
		MSSIM	0.1549	0.867	0.870	0.818	0.792	0.862	0.888	0.924	**0.941**
		SAM	37.072	5.228	5.174	6.834	6.819	4.725	4.208	**3.321**	3.390
	$\sigma = 0.3$	PSNR	10.46	27.85	28.33	27.48	26.73	28.23	29.90	31.84	**32.07**
		MSSIM	0.081	0.799	0.808	0.740	0.698	0.793	0.834	0.892	**0.911**
		SAM	48.258	6.581	6.490	8.653	8.232	6.448	5.332	**4.281**	4.439
Case 2	$\sigma = 1, \eta = 20$	PSNR	22.81	24.809	24.59	34.19	34.06	38.60	**40.59**	39.75	39.86
		MSSIM	0.743	0.785	0.790	0.956	0.955	0.985	**0.990**	0.985	0.985
		SAM	15.515	12.392	12.663	3.829	3.956	2.074	**1.685**	1.876	1.808
Case 3	$\sigma \sim \mathcal{U}(0.1, 0.2)$	PSNR	16.52	24.87	29.49	30.59	30.37	33.52	34.57	35.64	**35.87**
		MSSIM	0.2475	0.594	0.803	0.853	0.844	0.906	0.925	0.942	**0.959**
		SAM	29.517	11.940	6.731	5.787	5.890	3.705	3.264	2.819	**2.816**

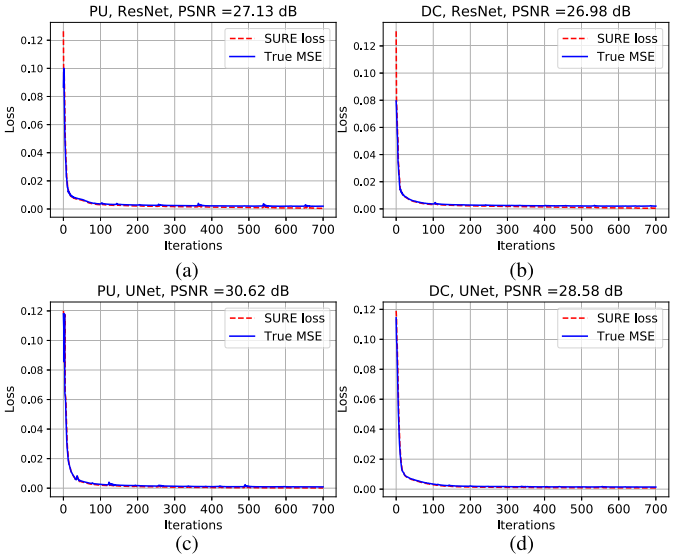


Fig. 5. Training loss and true MSE of the ResNet and UNet for denoising Case 1 with $\sigma = 0.3$. (a) ResNet, PU data set. (b) ResNet, DC data set. (c) Unet, PU data set. (d) Unet, DC data set.

(LRTA) [19], tensor learning dictionary (TDL) [20], an HSI denoising method using the first-order spectral roughness penalty in the wavelet domain (FORPDN) [11], the nonlocal transform-domain filter for volumetric data (BM4D) [10], an HSI denoising method via noise-adjusted iterative low-rank matrix approximation (NAILRMA) [22], an HSI restoration technique using the sparse and low-rank model (HyRes) [24], and the fast HSI denoising and inpainting based on low-rank and sparse representation (FastHyDe) [25]. All the competitive methods are run on the simulated PU and DC data sets using the parameters recommended in the corresponding papers. The following parameters are used in SURE-CNN: The number of conv-relu blocks is $K = 5$, the number of filters in the skip connection layers is $F = 5$, and the noise standard deviation $\widehat{\sigma}_i$ is estimated by (4). Tables II and III give the denoising results in terms of PSNR, MSSIM, and SAM for the DC and PU data sets, respectively. The obtained PSNR and MSSIM of SURE-CNN are higher than other methods in most cases. For high noise levels, such as in Case 1 with $\sigma \in \{0.3,\ 0.2,\ 0.1\}$ and in Case 3, SURE-CNN outperforms all the competitive methods. The margins are nearly 0.4 dB of PSNR, and a few percents of MSSIM in comparison with the second-best method, FastHyDe. However, FastHyDe gives the best PSNR in Case 1 with $\sigma = 0.05$ for both DC and PU data sets; and HyRes gives the best PSNR and MSSIM in Case 2 for the DC data set.

We choose $\sigma = 0.3$ for Case 1 and use the setting specified above for Cases 2 and 3 to demonstrate visual quality. Figs. 6–8 show the denoised results of a subscene of band 60 of the DC data set, while Figs. 9–11 show the denoised results of a subscene of band 60 of the PU data set. LRTA, TDL, FORPDN, and NAIRLMA cannot completely remove the noise as there is still noise in the denoised images. BM4D gives oversmooth denoised images. HyRes, FastHyDe, and SURE-CNN work well in filtering out the noise in all cases and provide clean denoised images. However, for denoising PU data set Case 1 (see Fig. 9) and Case 2 (see Fig. 10), HyRes is slightly worse than FastHyDe and SURE-CNN.

TABLE III

DENOISING FOR THE PU DATA SET USING DIFFERENT METHODS. THE EVALUATED METRICS ARE PSNR IN dB, AND MSSIM AND SAM IN DEGREES. THE RESULTS ARE THE AVERAGE VALUES OVER TEN RUNS. THE STANDARD DEVIATIONS FOR PSNR, MSSIM, AND SAM IN EACH METHOD ARE LESS THAN 0.06 dB, 0.0006, AND 0.01°, RESPECTIVELY. BEST RESULTS ARE IN BOLDFACE

PU	Noise level	Metric	Noisy	LRTA	TDL	FORPDN	BM4D	NAILRMA	HyRes	FastHyDe	SURE-CNN
Case 1	$\sigma = 0.05$	PSNR	26.02	35.67	39.24	37.84	38.01	38.46	38.75	**39.62**	39.17
		MSSIM	0.593	0.922	0.967	0.955	0.957	0.959	0.963	**0.970**	0.969
		SAM	16.584	5.172	3.287	3.934	3.938	3.752	3.541	3.185	**3.149**
	$\sigma = 0.1$	PSNR	20.00	32.43	35.46	34.26	34.35	34.65	35.24	36.75	**37.12**
		MSSIM	0.320	0.863	0.935	0.904	0.914	0.910	0.925	0.950	**0.955**
		SAM	29.683	6.462	4.559	5.536	5.495	5.336	4.845	4.053	**3.643**
	$\sigma = 0.2$	PSNR	13.98	28.81	31.68	30.33	30.39	30.20	31.67	33.62	**33.91**
		MSSIM	0.125	0.741	0.868	0.795	0.815	0.800	0.852	0.913	**0.918**
		SAM	47.376	7.947	5.952	7.915	7.892	7.689	6.571	5.287	**4.695**
	$\sigma = 0.3$	PSNR	10.46	26.53	29.38	27.86	28.25	27.47	29.69	31.75	**31.91**
		MSSIM	0.062	0.629	0.799	0.689	0.723	0.694	0.790	**0.878**	**0.878**
		SAM	57.886	9.401	6.937	9.670	9.545	9.501	7.520	6.334	**5.769**
Case 2	$\sigma = 1, \eta = 20$	PSNR	20.13	25.67	26.81	35.06	34.83	35.49	36.32	37.19	**37.66**
		MSSIM	0.403	0.612	0.699	0.920	0.921	0.926	0.940	0.952	**0.957**
		SAM	29.304	16.609	14.081	5.326	5.235	4.875	4.284	3.916	**3.593**
Case 3	$\sigma \sim \mathcal{U}(0.1, 0.2)$	PSNR	16.63	25.36	28.38	31.91	32.13	32.40	33.54	34.98	**35.67**
		MSSIM	0.212	0.573	0.740	0.855	0.867	0.866	0.899	0.932	**0.941**
		SAM	39.816	16.371	11.497	6.840	7.272	6.448	5.766	4.755	**4.269**

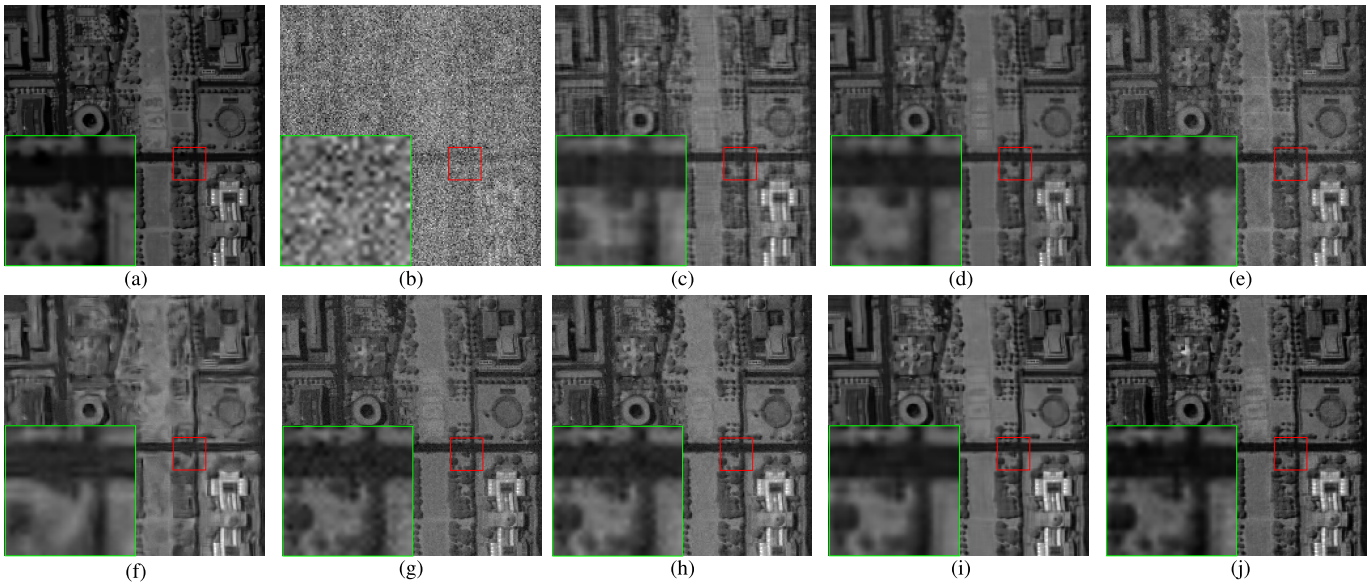


Fig. 6. Denoising for the DC data set band 60, Case 1 with $\sigma = 0.3$, using different methods. The green square is a zoomed-in area shown in the red square. (a) Reference image. (b) Noisy image. (c) LRTA. (d) TDL. (e) FORPDN. (f) BM4D. (g) NAILRMA. (h) HyRes. (i) FastHyDe. (j) SURE-CNN.

D. Performance: Real Data Sets

In this section, we show the denoising results using SURE-CNN and the competitive methods for two real data sets: the IP and UB data sets. Again, all the competitive methods use the default parameters. SURE-CNN uses the same parameters as in denoising of the simulated data, but we run it 500 iterations for the IP data set and 150 iterations for the UB data set. Fig. 12 shows the denoising results for the IP data set. Since the dominant noise in the IP data set is the Gaussian noise, all the denoising methods significantly remove the noise and improve the visual quality of the denoised images. However, FORPDN produces artifacts. The denoised images by BM4D are oversmooth. For the strongly contaminated noise band, such as band 104, only SURE-CNN can recover the image from noise, while all the other methods fail to remove the noise.

The denoised UB data set is shown in Fig. 13. It is harder to remove the noise in the UB data set because it contains mixed

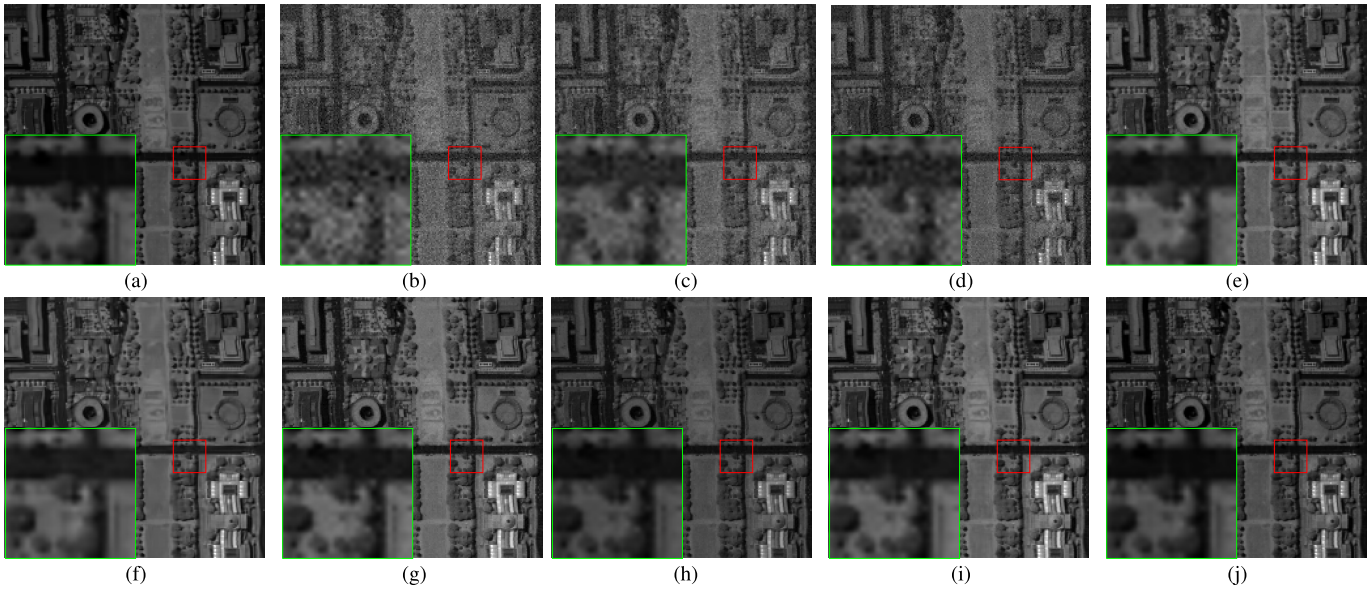


Fig. 7. Denoising for the DC data set band 60, Case 2 with $\sigma = 1, \eta = 20$, using different methods. The green square is a zoomed-in area shown in the red square. (a) Reference image. (b) Noisy image. (c) LRTA. (d) TDL. (e) FORPDN. (f) BM4D. (g) NAILRMA. (h) HyRes. (i) FastHyDe. (j) SURE-CNN.

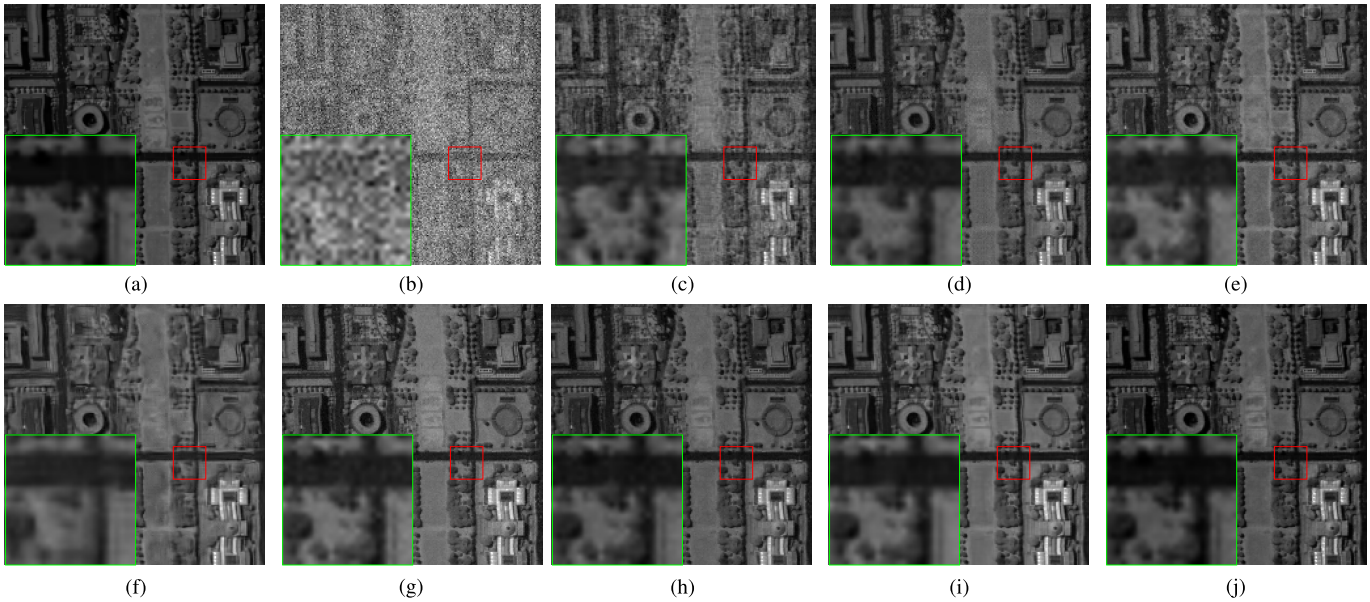


Fig. 8. Denoising for the DC data set band 60, Case 3 with $\sigma \sim \mathcal{U}(0.1, 0.2)$, using different methods. The green square is a zoomed-in area shown in the red square. (a) Reference image. (b) Noisy image. (c) LRTA. (d) TDL. (e) FORPDN. (f) BM4D. (g) NAILRMA. (h) HyRes. (i) FastHyDe. (j) SURE-CNN.

types of Gaussian and stripped noises. All the competitive methods can filter out the Gaussian noise, but they are unable to remove the stripped noise. It is easy to recognize that SURE-CNN can separate the images from all types of noises. Especially, for very strongly noise affected bands, such as bands 144 and 208, SURE-CNN yields very clean denoised images, in contrast to the competitive methods. It is also noticed that, for bands 108 and 139, which have low noise levels, FORPDN and NAILRMA also work well.

E. Performance: Subspace SURE-CNN

Here, we evaluate the performance of SURE-CNN in a subspace found by SVD. We use the simulated PU and DC data sets in Case 1 with $\sigma = 0.3$ for these experiments. The evaluated metrics are PSNR in decibels and running time in seconds. All the parameters of SURE-CNN are selected as in Section IV-B. The optimal training iterations depend on data and denoising cases. For the denoising Case 1 with $\sigma = 0.3$, the denoising process is run over 500 iterations. We run SURE-CNN using a Linux computer equipped with an Nvidia TitanX GPU of 12-GB memory, under the Tensorflow 2.0 GPU environment. Fig. 14 shows the PSNR and running time of SURE-CNN in different subspace dimensions and the full-rank dimension (i.e., do not use subspace). It is shown that the subspace SURE-CNN not only reduces the running time but also improves the PSNR over the full-rank HSI, such

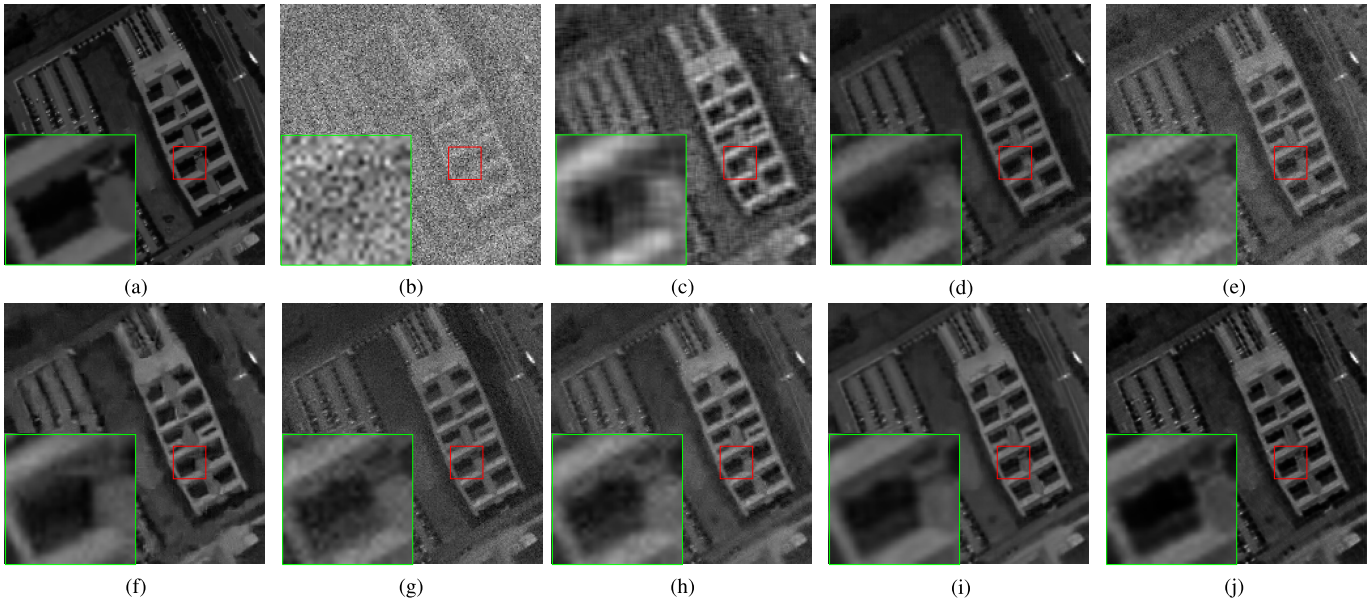


Fig. 9. Denoising for the PU data set band 60, Case 1 with $\sigma = 0.3$, using different methods. The green square is a zoomed-in area shown in the red square. (a) Reference image. (b) Noisy image. (c) LRTA. (d) TDL. (e) FORPDN. (f) BM4D. (g) NAILRMA. (h) HyRes. (i) FastHyDe. (j) SURE-CNN.

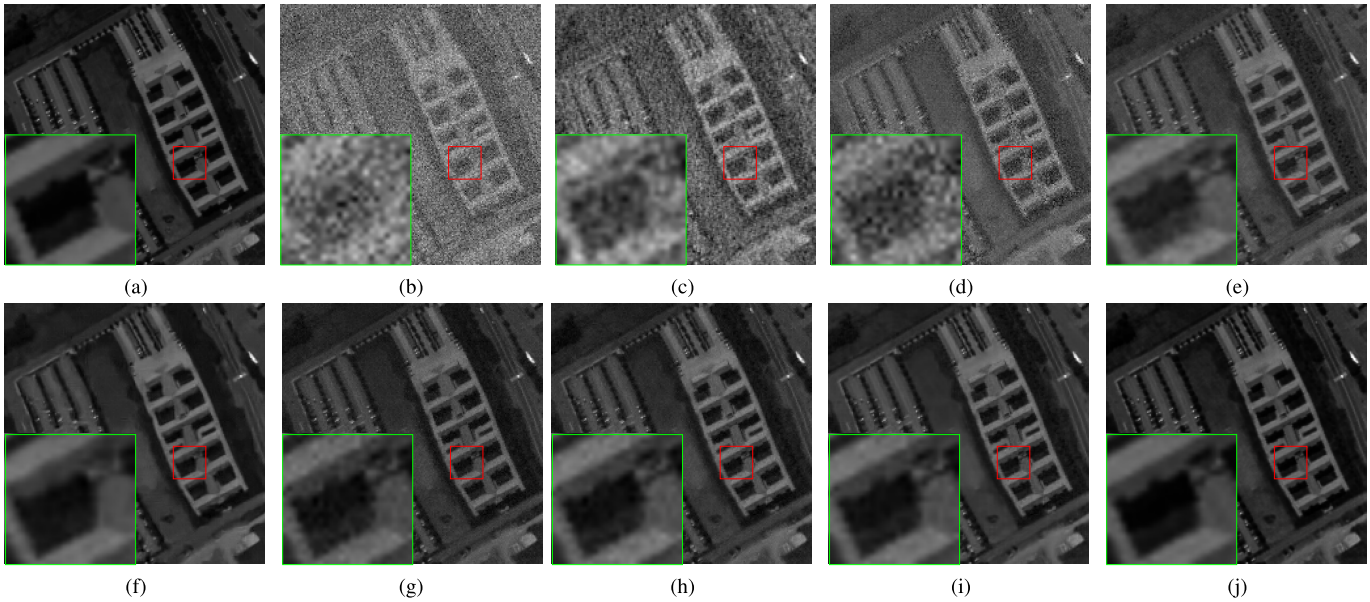


Fig. 10. Denoising for the PU data set band 60, Case 2 with $\sigma = 1, \eta = 20$, using different methods. The green square is a zoomed-in area shown in the red square. (a) Reference image. (b) Noisy image. (c) LRTA. (d) TDL. (e) FORPDN. (f) BM4D. (g) NAILRMA. (h) HyRes. (i) FastHyDe. (j) SURE-CNN.

as $k_{\text{sub}} = 5$ gives PNSR = 32.39 dB for the PU data set and $k_{\text{sub}} = 10$ gives PSNR = 32.65 dB for the DC data set. The results are shown in Fig. 15 for bands 60 of both data sets. This is because the low-rank approximation using SVD can be considered as a predenoising technique, and it can benefit the final denoising performance.

F. Further Discussion

In this section, we further discuss the comparison performance between SURE-CNN and two other DL-based methods. We also show the running time of SURE-CNN against competitive methods.

1) Comparison SURE-CNN With the DL-Based Methods: Here, we compare SURE-CNN with two DL-based denoising methods. The first method is a supervised DL denoising method, HSI-SDeCNN, proposed in [31]. The second method is an unsupervised DL denoising method, DIP-HSI, proposed in [36]. HSI-SDeCNN used a CNN that was trained using small noisy-clean image patches extracted from the synthesis DC data set. As shown in [31], HSI-SDeCNN concerned the noisy simulated data set only as in Case 1 with various values of σ. It is usually time-consuming to train a supervised DL network. Thus, we use a pretrained HSI-SDeCNN network that was provided by

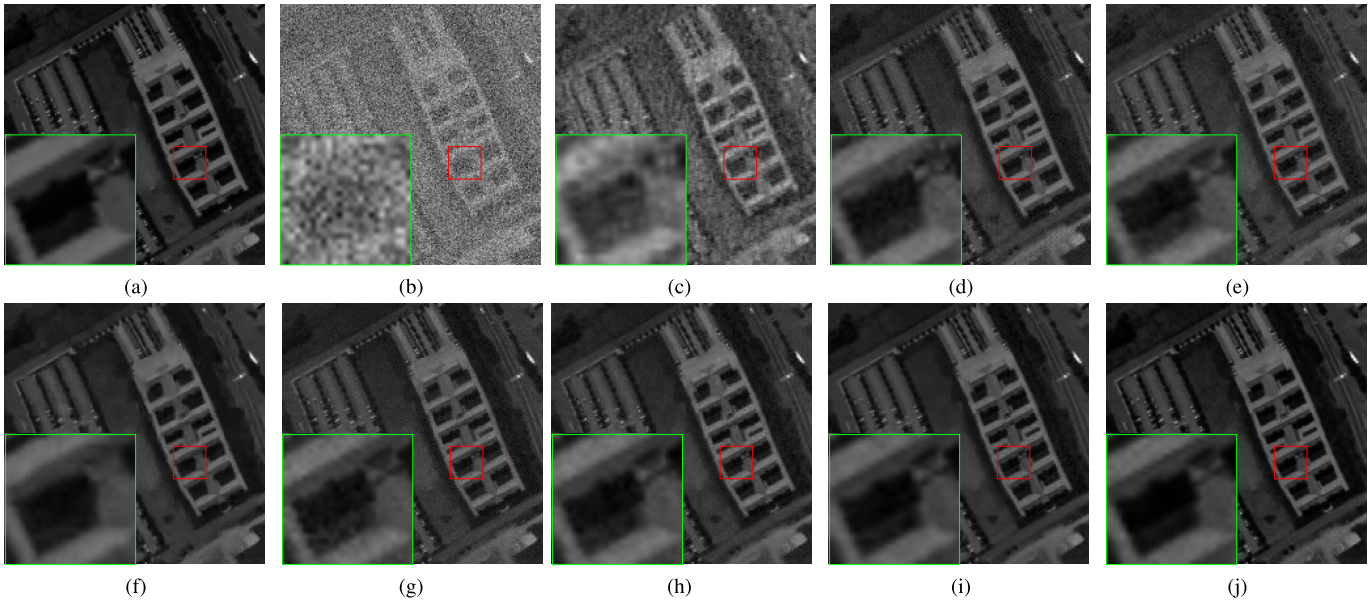


Fig. 11. Denoising for the PU data set band 60, Case 3 with $\sigma = \mathcal{U}(0.1, 0.2)$, using different methods. The green square is a zoomed-in area shown in the red square. (a) Reference image. (b) Noisy image. (c) LRTA. (d) TDL. (e) FORPDN. (f) BM4D. (g) NAILRMA. (h) HyRes. (i) FastHyDe. (j) SURE-CNN.

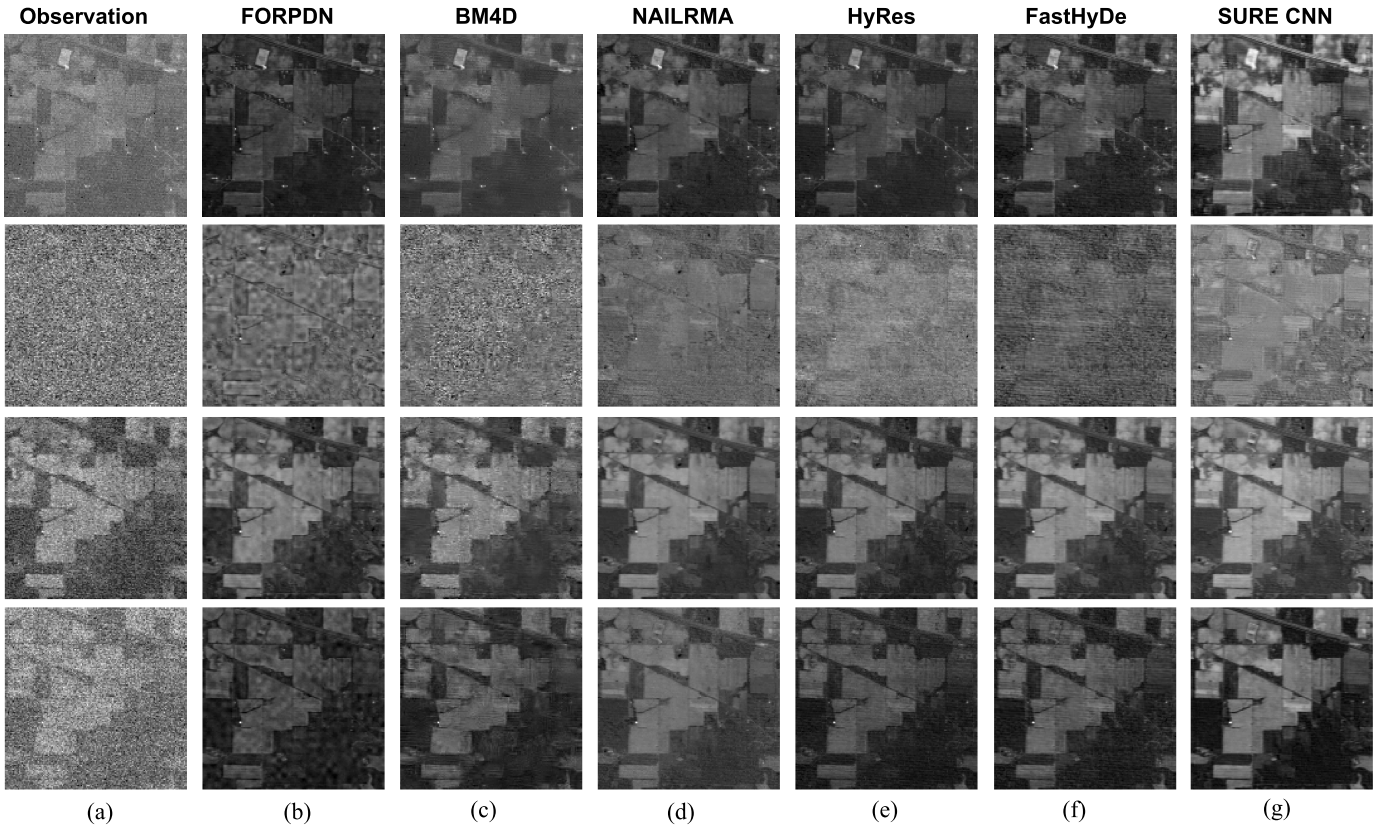


Fig. 12. Denoising for the IP data set using different methods, from top to bottom: bands 2, 104, 149, and 219.

Maffei *et al.* [31]. DIP-HSI uses default setting parameters. Our proposed method uses the same parameter as in Section IV-B. To be fair, all the methods are tested on the simulated data set that was used in [31]. It is a part of the DC data set that has 200 $\times$ 200 pixels of 191 bands. It is shown in false color using bands 57, 27, and 17 in the first column of Fig. 15.

Table IV and Figs. 16–18 show the denoising results using SURE-CNN and competitive DL-based methods. SURE-CNN outperforms HSI-SDeCNN and DIP-HSI in terms of PSNR

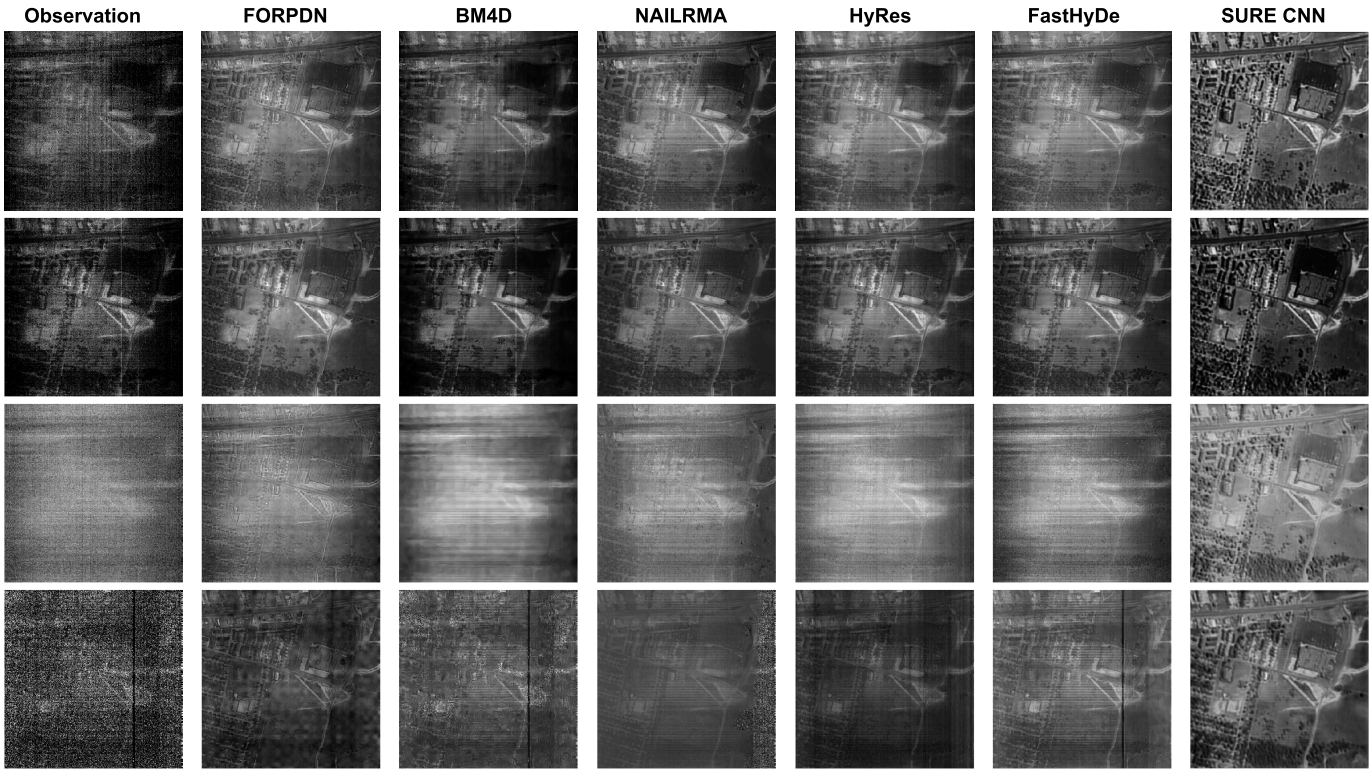
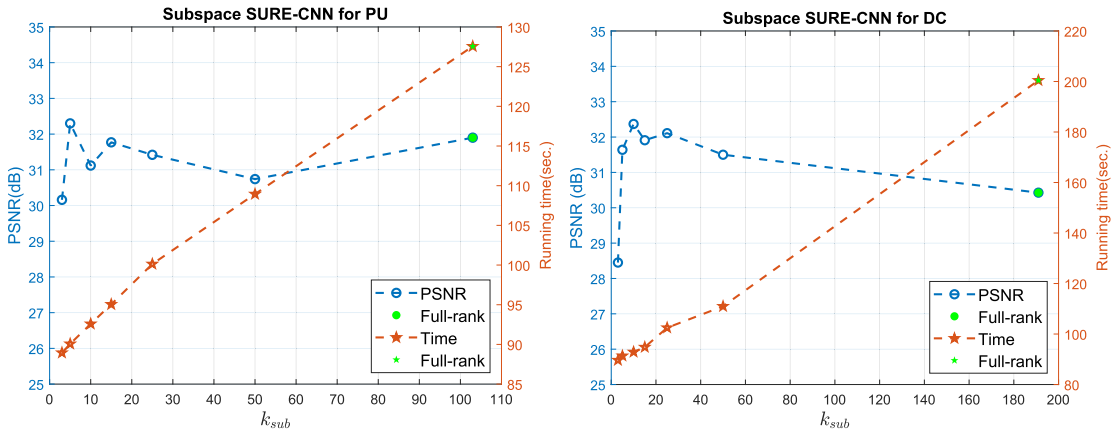
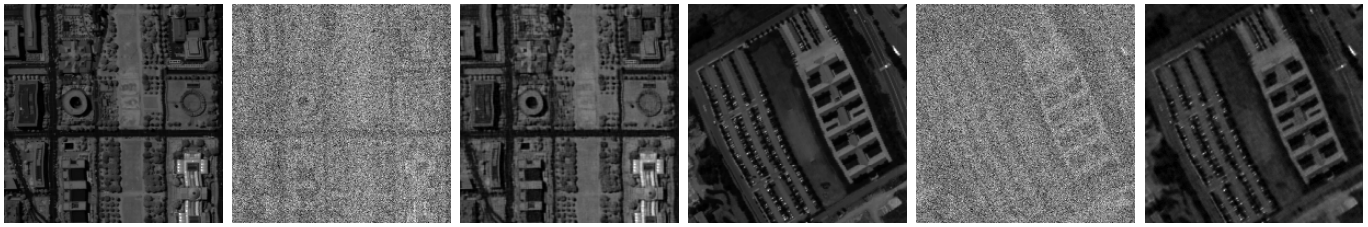


Fig. 13. Denoising for the UB data set using different methods, from top to bottom: bands 108, 139, 144, and 208.



(a) (b)

Fig. 14. SURE-CNN subspace for denoising Case 1 with $\sigma = 0.3$. The green markers show SURE-CNN in full-rank (no subspace). (a) PU data set. (b) DC data set.



(a) (b) (c) (e) (f)

Fig. 15. Denoising results using Subspace SURE-CNN for the DC and PU data sets, band 60. (a) Reference image (DC). (b) Noisy image (DC). (c) Subspace SURE-CNN ($k_{\text{sub}} = 10$). (d) Reference image (PU). (e) Noisy image (PU). (f) Subspace SURE-CNN ($k_{\text{sub}} = 5$).

and MSSIM. Also, the denoised images from SURE-CNN are significantly cleaner than the denoised images obtained by HSI-SDeCNN and DIP-HSI.

2) Running Time: We evaluate the running time (in seconds) for SURE-CNN and the competitive methods applied to the real IP and UB data sets with the experiments described in

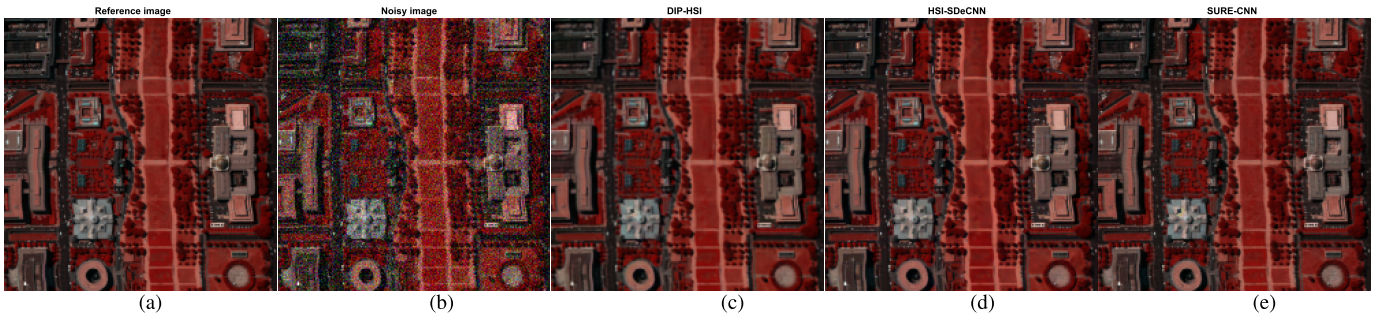


Fig. 16. Denoising for a test data set Case 1 with $\sigma = 25/255$ using DL-based methods. (a) Reference image. (b) Noisy image. (c) DIP-HSI. (d) HSI-SDeCNN. (e) SURE-CNN.

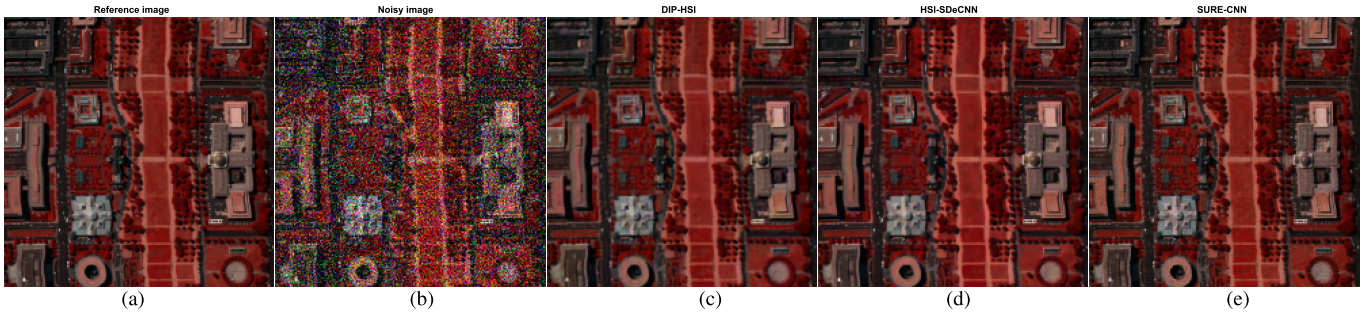


Fig. 17. Denoising for a test data set Case 1 with $\sigma = 50/255$ using DL-based methods. (a) Reference image. (b) Noisy image. (c) DIP-HSI. (d) HSI-SDeCNN. (e) SURE-CNN.

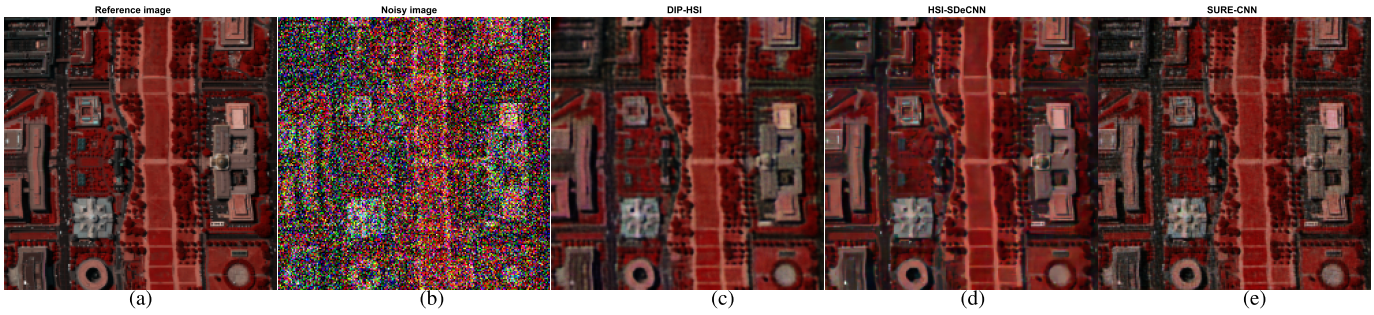


Fig. 18. Denoising for a test data set Case 1 with $\sigma = 100/255$ using DL-based methods. (a) Reference image. (b) Noisy image. (c) DIP-HSI. (d) HSI-SDeCNN. (e) SURE-CNN.

TABLE IV

DENOISING RESULTS GIVEN BY PSNR IN dB AND MSSIM USING DL-BASED METHODS. THE RESULTS ARE THE AVERAGE VALUES OVER TEN RUNS. THE STANDARD DEVIATIONS FOR PSRN AND MSSIM IN EACH METHOD ARE LESS THAN 0.06 dB AND 0.0005, RESPECTIVELY. THE BEST RESULTS ARE SHOWN IN BOLDFACE

Noise level	Metric	Noisy	HSI-SDeCNN	DIP-HSI	SURE-CNN
$\sigma = 25/255$	PSNR	20.17	33.18	34.88	**36.07**
	MSSIM	0.4910	0.9626	0.9494	**0.9746**
$\sigma = 50/255$	PSNR	14.15	31.03	31.20	**33.23**
	MSSIM	0.2375	0.9178	0.9243	**0.9527**
$\sigma = 100/255$	PSNR	8.13	27.65	27.71	**28.98**
	MSSIM	0.0804	0.8522	0.8245	**0.8881**

Section IV-D repeated exactly. Here, SURE-CNN, which is implemented using Tensorflow 2.0 GPU, runs on a Linux computer with Nvidia Titan X GPU of 12-GB memory.

TABLE V

AVERAGE RUNNING TIME (IN SECONDS) OVER FIVE RUNS FOR DIFFERENT DENOISING METHODS. THE STANDARD DEVIATIONS OF SURE-CNN ARE 1.54 AND 2.30 S FOR THE IP AND UB DATA SETS, RESPECTIVELY

Dataset	FORPDN	BM4D	NAIL-RMA	HyRes	FastHyDe	SURE-CNN
IP	2.08	281.21	94.09	1.19	**0.15**	89.87
UB	5.79	1237.7	362.44	3.99	**0.79**	126.42

The remaining denoising methods are implemented using MATLAB R2019b, run on a Linux computer with eight-core Intel CPU 3.2 GHz and 64-GB RAM. The running times are given in Table V. The fastest method is FastHyDe with less than 1 s for denoising the IP and UB data sets, while BM4D is the slowest method. The average running time of SURE-CNN is in the middle between the slow group methods (BM4D and NAILRMA) and the fast group methods (FastHyDe, HyRes,

and FORPDN). SURE-CNN running time is 89.87 s for the IP data set and 126.42 s for the UB data set.

V. Conclusion

We have proposed a new SURE-based CNN method for HSI denoising. The proposed HSI SURE loss function approximates the true MSE very well. Thus, using the SURE loss for CNN can significantly prevent overfitting. The experimental results verify that our proposed method improves the denoising performance in both simulated noisy and real noisy data sets. We also proposed a subspace SURE-CNN method that enables SURE-CNN works in reduced-rank HSI data. This mechanism significantly reduces the running time and may improve denoising performance in some cases.

Acknowledgment

The authors would like to thank an Associate Editor and the anonymous reviewers for their comments and suggestions, which greatly helped to improve the technical quality and presentation of this article.

References

[1] B. Zhang, D. Wu, L. Zhang, Q. Jiao, and Q. Li, "Application of hyperspectral remote sensing for environment monitoring in mining areas," *Environ. Earth Sci.*, vol. 65, no. 3, pp. 649–658, Feb. 2012.

[2] J. C. Harsanyi and C.-I. Chang, "Hyperspectral image classification and dimensionality reduction: An orthogonal subspace projection approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 32, no. 4, pp. 779–785, Jul. 1994.

[3] D. Manolakis and G. Shaw, "Detection algorithms for hyperspectral imaging applications," *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 29–43, Jan. 2002.

[4] Q. Zhang, Q. Yuan, J. Li, X. Liu, H. Shen, and L. Zhang, "Hybrid noise removal in hyperspectral imagery with a spatial–spectral gradient network," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 10, pp. 7317–7329, Oct. 2019.

[5] 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.

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

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

[8] I. Atkinson, F. Kamalabadi, and D. L. Jones, "Wavelet-based hyperspectral image estimation," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, vol. 2, Jul. 2003, pp. 743–745.

[9] B. Rasti, J. R. Sveinsson, M. O. Ulfarsson, and J. A. Benediktsson, "Hyperspectral image denoising using 3D wavelets," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2012, pp. 1349–1352.

[10] M. Maggioni, V. Katkovnik, K. Egiazarian, and A. Foi, "Nonlocal transform-domain filter for volumetric data denoising and reconstruction," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 119–133, Jan. 2013.

[11] B. Rasti, J. R. Sveinsson, M. O. Ulfarsson, and J. A. Benediktsson, "Hyperspectral image denoising using first order spectral roughness penalty in wavelet domain," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 6, pp. 2458–2467, Jun. 2014.

[12] Q. Yuan, L. Zhang, and H. Shen, "Hyperspectral image denoising employing a spatial–spectral adaptive total variation model," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 10, pp. 3660–3677, Oct. 2012.

[13] H. Zhang, "Hyperspectral image denoising with cubic total variation model," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. I-7, pp. 95–98, Jul. 2012.

[14] Y. Chang, L. Yan, H. Fang, and C. Luo, "Anisotropic spectral–spatial total variation model for multispectral remote sensing image destriping," *IEEE Trans. Image Process.*, vol. 24, no. 6, pp. 1852–1866, Jun. 2015.

[15] H. Kumar Aggarwal and A. Majumdar, "Hyperspectral image denoising using spatio-spectral total variation," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 3, pp. 442–446, Mar. 2016.

[16] T. Lu, S. Li, L. Fang, Y. Ma, and J. A. Benediktsson, "Spatial–spectral adaptive sparse representation for hyperspectral image denoising," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 1, pp. 373–385, Jan. 2016.

[17] J. Li, Q. Yuan, H. Shen, and L. Zhang, "Noise removal from hyperspectral image with joint spatial–spectral distributed sparse representation," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 9, pp. 5425–5439, Sep. 2016.

[18] X. Bai, F. Xu, L. Zhou, Y. Xing, L. Bai, and J. Zhou, "Nonlocal similarity based nonnegative tucker decomposition for hyperspectral image denoising," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 3, pp. 701–712, Mar. 2018.

[19] N. Renard, S. Bourennane, and J. Blanc-Talon, "Denoising and dimensionality reduction using multilinear tools for hyperspectral images," *IEEE Geosci. Remote Sens. Lett.*, vol. 5, no. 2, pp. 138–142, Apr. 2008.

[20] Y. Peng, D. Meng, Z. Xu, C. Gao, Y. Yang, and B. Zhang, "Decomposable nonlocal tensor dictionary learning for multispectral image denoising," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2949–2956.

[21] H. Zhang, W. He, L. Zhang, H. Shen, and Q. Yuan, "Hyperspectral image restoration using low-rank matrix recovery," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 8, pp. 4729–4743, Aug. 2014.

[22] W. He, H. Zhang, L. Zhang, and H. Shen, "Hyperspectral image denoising via noise-adjusted iterative low-rank matrix approximation," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 3050–3061, Jun. 2015.

[23] Y. Zhao and J. Yang, "Hyperspectral image denoising via sparsity and low rank," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2013, pp. 1091–1094.

[24] B. Rasti, M. O. Ulfarsson, and P. Ghamisi, "Automatic hyperspectral image restoration using sparse and low-rank modeling," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 12, pp. 2335–2339, Dec. 2017.

[25] L. Zhuang and J. M. Bioucas-Dias, "Fast hyperspectral image denoising and inpainting based on low-rank and sparse representations," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 3, pp. 730–742, Mar. 2018.

[26] G. Ma, T.-Z. Huang, J. Huang, and C.-C. Zheng, "Local low-rank and sparse representation for hyperspectral image denoising," *IEEE Access*, vol. 7, pp. 79850–79865, 2019.

[27] 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.

[28] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep CNN denoiser prior for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 3929–3938.

[29] T. Brooks, B. Mildenhall, T. Xue, J. Chen, D. Sharlet, and J. T. Barron, "Unprocessing images for learned raw denoising," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 11036–11045.

[30] Q. Yuan, Q. Zhang, J. Li, H. Shen, and L. Zhang, "Hyperspectral image denoising employing a spatial–spectral deep residual convolutional neural network," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 1205–1218, Feb. 2019.

[31] A. Maffei, J. M. Haut, M. E. Paoletti, J. Plaza, L. Bruzzone, and A. Plaza, "A single model CNN for hyperspectral image denoising," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 4, pp. 2516–2529, Apr. 2019.

[32] W. Xie and Y. Li, "Hyperspectral imagery denoising by deep learning with trainable nonlinearity function," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 11, pp. 1963–1967, Nov. 2017.

[33] Y. Chang, L. Yan, H. Fang, S. Zhong, and W. Liao, "HSI-DeNet: Hyperspectral image restoration via convolutional neural network," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 667–682, Feb. 2019.

[34] W. Liu and J. Lee, "A 3-D atrous convolution neural network for hyperspectral image denoising," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 8, pp. 5701–5715, Aug. 2019.

[35] Q. Zhang, Q. Yuan, J. Li, F. Sun, and L. Zhang, "Deep spatio-spectral Bayesian posterior for hyperspectral image non-i.i.d. noise removal," *ISPRS J. Photogramm. Remote Sens.*, vol. 164, pp. 125–137, Jun. 2020.

[36] O. Sidorov and J. Y. Hardeberg, "Deep hyperspectral prior: Single-image denoising, inpainting, super-resolution," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Oct. 2019, pp. 3844–3851.

[37] R. Imamura, T. Itasaka, and M. Okuda, "Zero-shot hyperspectral image denoising with separable image prior," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2019, pp. 1416–1420.

[38] C. M. Stein, "Estimation of the mean of a multivariate normal distribution," *Ann. Statist.*, vol. 9, no. 6, pp. 1135–1151, Nov. 1981.
[39] V. Solo, "A sure-fired way to choose smoothing parameters in ill-conditioned inverse problems," in *Proc. 3rd IEEE Int. Conf. Image Process. (ICIP)*, Lausanne, Switzerland, vol. 3, 1996, pp. 89–92.
[40] A. Signoroni, M. Savardi, A. Baronio, and S. Benini, "Deep learning meets hyperspectral image analysis: A multidisciplinary review," *J. Imag.*, vol. 5, no. 5, p. 52, May 2019.
[41] N. Audebert, B. Le Saux, and S. Lefèvre, "Deep learning for classification of hyperspectral data: A comparative review," *IEEE Geosci. Remote Sens. Mag.*, vol. 7, no. 2, pp. 159–173, Jun. 2019.
[42] J. Lehtinen *et al.*, "Noise2Noise: Learning image restoration without clean data," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, Jul. 2018, pp. 2965–2974.
[43] A. Krull, T.-O. Buchholz, and F. Jug, "Noise2 void—Learning denoising from single noisy images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2129–2137.
[44] V. Lempitsky, A. Vedaldi, and D. Ulyanov, "Deep image prior," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9446–9454.
[45] J. Liu, Y. Sun, X. Xu, and U. S. Kamilov, "Image restoration using total variation regularized deep image prior," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2019, pp. 7715–7719.
[46] G. Mataev, P. Milanfar, and M. Elad, "DeepRED: Deep image prior powered by RED," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Oct. 2019. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9022249 and https://openaccess.thecvf.com/ICCV2019_workshops/ICCV2019_LCI
[47] C. A. Metzler, A. Mousavi, R. Heckel, and R. G. Baraniuk, "Unsupervised learning with Stein's unbiased risk estimator," in *Proc. Int. Biomed. Astronomical Signal Process. Frontiers Workshop*, Feb. 2019, pp. 67–68.
[48] S. Soltanayev and S. Y. Chun, "Training deep learning based denoisers without ground truth data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 3257–3267.
[49] M. Zhussip, S. Soltanayev, and S. Y. Chun, "Training deep learning based image denoisers from undersampled measurements without ground truth and without image prior," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 10255–10264.
[50] 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.
[51] J. M. Bioucas-Dias and J. M. P. Nascimento, "Hyperspectral subspace identification," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 8, pp. 2435–2445, Aug. 2008.
[52] S. Mallat, *A Wavelet Tour of Signal Processing*. New York, NY, USA: Academic, 1998.
[53] F. J. Anscombe, "The transformation of Poisson, binomial and negative-binomial data," *Biometrika*, vol. 35, nos. 3–4, pp. 246–254, 1948.
[54] H. V. Nguyen, M. O. Ulfarsson, and J. R. Sveinsson, "Sure based convolutional neural networks for hyperspectral image denoising," in *Proc. IEEE Geosci. Remote Sens. Symp.*, 2020.
[55] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
[56] X. Mao, C. Shen, and Y.-B. Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2802–2810.
[57] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3431–3440.
[58] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput.-Assist. Intervent*. Springer, 2015, pp. 234–241.
[59] 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.
[60] F. Palsson, J. Sveinsson, and M. Ulfarsson, "Sentinel-2 image fusion using a deep residual network," *Remote Sens.*, vol. 10, no. 8, p. 1290, Aug. 2018.

Han V. Nguyen (Graduate Student Member, IEEE) received the B.S. degree in information and communication engineering from the Hanoi University of Science and Technology, Hanoi, Vietnam, in 2008, and the M.S. degree in information and communication engineering from Chosun University, Gwangju, South Korea, in 2014. He is pursuing the Ph.D. degree in electrical and computer engineering with the University of Iceland, Reykjavik, Iceland.

In 2010, he joined the Faculty of Electrical and Electronic Engineering, Nha Trang University, Nha Trang, Vietnam, as a Lecturer. His research interests include digital image processing, remote sensing, machine learning, and deep learning.

Magnus O. Ulfarsson (Senior Member, IEEE) received the B.S. and M.S. degrees from the University of Iceland, Reykjavik, Iceland, in 2002, and the Ph.D. degree from the University of Michigan, Ann Arbor, MI, USA, in 2007.

In 2007, he joined the University of Iceland, where he is a Professor and the Chair of the Faculty of Electrical and Computer Engineering. Since 2013, he has been with deCODE Genetics, Reykjavik. His research interests include statistical signal processing, image processing, machine learning, remote sensing, medical imaging, and genomics.

Dr. Ulfarsson has been an Associate Editor of the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING since 2018.

Johannes R. Sveinsson (Senior Member, IEEE) received the B.S. degree from the University of Iceland, Reykjavik, South Korea, and the M.S. and Ph.D. degrees from Queen's University, Kingston, ON, Canada, all in electrical engineering.

He was with the Laboratory of Information Technology and Signal Processing from 1981 to 1982 and the Engineering Research Institute and the Faculty of Electrical and Computer Engineering, University of Iceland, as a Senior Member of Research Staff and a Lecturer, respectively, from 1991 to 1998. He was a Visiting Research Student with the Imperial College of Science and Technology, London, U.K., from 1985 to 1986. At Queen's University, he held teaching and research assistantships. He was the Chair of the Faculty of Electrical and Computer Engineering, University of Iceland, from 2000 to 2006 and from 2010 to 2014, and a member of the University Council, University of Iceland, from 2002 to 2006. He is a Professor with the Faculty of Electrical and Computer Engineering, University of Iceland. His research interest includes systems and signal theory.

Dr. Sveinsson received the Queen's Graduate Awards from Queen's University. He was a recipient of the Best Paper Award Novel Engineering Application at ANNIE 1995 and a co-recipient of the 2013 IEEE Geoscience and Remote Sensing Society Highest Impact Paper Award. He has been an Associate Editor of *Remote Sensing* since 2018.