The Hong Kong Polytechnic University

Department of Computing

Patch Group Based Nonlocal Self-Similarity Prior Modeling for Image Denoising

Jun Xu

A thesis
submitted in partial fulfilment of the requirements
for the degree of

Doctor of Philosophy

August 24, 2017

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____ (Signed)

Jun Xu (Name of student)

Abstract

The nonlocal self-similarity (NSS) prior of natural images has been extensively studied in many image restoration methods. In this thesis, we exploit the NSS property of external natural images, external guided internal NSS property, and internal NSS property for image denoising tasks.

Acknowledgement

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Contents

List of Figures

List of Tables

1 Introduction

Nowadays, CCD or CMOS cameras are becoming more and more important in many aspects of human life such as photography, artificial intelligence, and security system. For each camera product, the camera imaging pipeline in the camera is of particular importance since it is the core part to transform the photons reflected by the real scene being captured in the camera sensor into the pixel values of an image, which can be displayed on a screen. During the camera imaging process, the noise is unavoidablely generated due to many reasons. Two major reasons of noise generation are the discrete nature of light and the thermal agitation, which can cause the photon shot noise and the dark-current noise, respectively. Image denoising is the problem of recovering the laten clean image from the captured noise version.

Chapter abstract This chapter will introduce the image noise and its acquirezation equation, the image denoising problem, the objective measures to evaluate the image denoising performance, the proposed denoising methods. Finally, I will summarize the structural of this thesis, and the contribution I made in this thesis.

1.1 The Image Noise Formulation

The realistic noise are very complex in real photography images captured by the camera sensors. One major reason is that the noise is unable to be explicitly modeled by some definitive probabilistic distributions. Often the sRGB images we look at the screen have been processed via the camera imaging pipeline, during which the noise will be much more complex than its initial status.

The noise are generated mainly due to the discrete nature of light and the thermal agitation, which will cause unstable measurement in camera sensors. The major types of noise generated during the imaging pipeline are the random noise, the spatial non-uniformity noise, and quantization noise. The random noise includes photon shot noise, dark current, and readout noise. The spatial non-uniformity noise includes the fixed pattern noise (PRNU, DCNU), CCD/CMOS specific noise.

To better describe the noise quantitatively, we provide a simplified signal acquisition model [] including various noise sources (for each pixel) as follows:

$$\boldsymbol{P} = f((g_{cv}(\boldsymbol{C} + \boldsymbol{D}) + \boldsymbol{N}_{reset})g_{out} + \boldsymbol{N}_{out}) + \boldsymbol{Q}. \tag{1.1}$$

The above equation is explained in details:

- $\boldsymbol{P}$ is the raw pixel value;
- f is the camera response function, usually a linear function before attaining a saturation threshold;
- $\boldsymbol{C}$ is the number of absorbed electrons (charges) transformed from the photons via the photon-diodes in the camera sensor, which can be modeled by a Poisson distribution;
- $\boldsymbol{D}$ is the number of absorbed electrons generated by dark current by thermal generation, which is often modeled by a Poisson distribution;
- $\boldsymbol{N}_{reset}$ is the thermal noise generated by the readout circuitry (or reset noise related to reset voltage), which can be well modeled by a Gaussian disribution;
- $\boldsymbol{N}_{out}$ is the readout noise, which is also modeled by a Gaussian distribution;
- $\boldsymbol{Q}$ is the quantization error happened during rounding to interger values, usually uniformly distributed and normally negligible compared to the readout noise;
- g_{cv} is the equivalent capacitance (EC) of the photo-diode and the gain factor during charge to voltage conversion;
- g_{out} is the gain factor during voltage to pixel value conversion (readout).

After some merging and simplifying, the acquisition model of the signal can be formulated as follows:

$$\begin{aligned} \mathbf{P} &= f((g_{cv}(\mathbf{C} + \mathbf{D}) + \mathbf{N}_{reset})g_{out} + \mathbf{N}_{out}) + \mathbf{Q}, \\ &= f(g_{cv}g_{out}(\mathbf{C} + \mathbf{D}) + g_{out}\mathbf{N}_{reset} + \mathbf{N}_{out}) + \mathbf{Q}, \\ &= f(g\lambda + N_R) + \mathbf{Q}, \end{aligned} \tag{1.2}$$

where $g = g_{cv}g_{out}$ is the overall camera gain factor, $\lambda = \mathbf{C} + \mathbf{D}$ is # of electrons in pixel capacitor and also Poisson distributed, since "Sum of Independent Poisson Random Variables is also Poisson", and $N_R = g_{out}\mathbf{N}_{reset} + \mathbf{N}_{out}$ is the overall readout noise and also Gaussian distributed, since "Sum of Independent Gaussian Random Variables is also Gaussian". In summary, the overall noise before the camera imaging pipeline can be modeled by a mixed Poisson and Gaussian distribution []. Unfortunately, the realistic noise will become very complex after being processed during the camera imaging pipeline [**crosschannel**]. This makes the image denoising an important and challenging task.

To make the problem easier, in image denoising community, the most common tested noise is additive white Gaussian noise (AWGN) [**ksvd**]. In AWGN degraded image, each pixel is corrupted by a random value following Gaussian distribution with zero mean and a certain standard deviation (std). Note that for each pixel, the std of the Gaussian distribution is the same, and for all the pixels, the noise values are sampled independently. The AWGN noise is similar to the read-out noise in digital cameras.

In order to obtain images of good quality while still maintaining the structures and details of the captured scenes, image denoising is an essential step. In this thesis, I will present my work on image denoising, in which we focus on synthetic additive white Gaussian noise (AWGN) and the realistic noise in real-world images. The AWGN noise is described as a Gaussian distribution $\mathcal{N}(0, \sigma^2)$, which means that the noise is Gaussian distributed with 0 mean and σ standard deviation. Most of existing methods focus on processing AWGN noise since it is a perfect testing bed for evaluating the proposed methods as well as other image restoration problems such as image super-resolution, deblurring, inpainting, etc.

1.2 Image Denoising

In general, image denoising aims to recover the latent clean image $\boldsymbol{x}$ from the observed noisy image $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n}$, where $\boldsymbol{n}$ is assumed to be the additive noise. $\boldsymbol{n}$ is often assumed to be synthetic additive white Gaussian noise, or the realistic noise in real-world images. Image denoising can be viewed as a regression problem, in which a *plausible* clean image can be obtained from the infinite number of possible candidates. The word *plausible* means that

the denoised image should look like the noisy image but without the noise component.

Image denoising would be very hard if we do not employ any prior information on it. The reason is that we do not know what exactly the latent clean image is without the prior information of the clean image. Hence, it is meaningful to exploit the most *plausible* image under some prior information. The most commonly used prior information in image denoising community is the Bayesian rule, which is also known as maximum A-posterior (MAP) property. Under the MAP framework, the most *plausible* latent clean image is the one which has the maximum Bayesian probability given the given noisy version. The posteriro probability can be measured by some existing evaluation methods which I will introduce in the following sections. In fact, the probability or measurements can measure the closeness of the latent clean image to the given noisy image. The closeness is usually measured by the ℓ_2 norm of the difference between the two images mentioned above. There are many latent clean images with the same ℓ_2 norm distance with the given noisy image. But some images in the circle are more *plausible* than the others due to the aspects of less artifacts, better structural preservation, and less remaining noise, etc.

1.3 Evaluating Denoising Performance

In order to achieve the maximum Bayesian probability, we need calculate the measurements of goodness for the denoised images. A natural problem is, how to measure the quality of the denoised image? It is very important to find better answer to this question. In fact, the research of image quality accessment is to find good algorithms to measure the quality of images under different situations and applications including image denoising.

A initial understanding is that the answer to the above question is largely depends on the situations we face when we perform image denoising experiments. When we perform synthetic experiments on additive white Gaussian nosie (AWGN), we usually already have the original clean image, the noisy image is generated by adding synthetic AWGN noise to the clean image. Then we can directly measure the quality of the denoised image by some existing image quality assessment (IQA) metrics. When we do not have the clean image provided as “ground truth”, a possible and final solution is to measure

the image quality by relying on human subjective evaluation. The IQA metrics can be roughly divided into two major directions: 1) full reference IQA; and 2) no reference IQA. Full reference IQA metrics are based on the assumption that the true underlying image is available in order to compute a measure, while no reference IQA metrics perform quality assessments without the reference image since the true underlying image is not available.

RMSE and PSNR: We mentioned previously that the it is corresponding to the ℓ_2 norm or equally mean square error (MSE) which is used to measure the distance between the denoised image and the given noisy image. In fact, the MSE measure is closely related to the famous peak signal to noise ratio (PSNR) metric. PSNR is the most commonly used full reference IQA metric for many image restoration tasks including denoising. The definition of PSNR can be formulated as follows (for 8-bit image):

$$\text{PSNR} = 20\log_{10}(\frac{2^8}{\text{RMSE}(\boldsymbol{x}, \boldsymbol{y})}), \tag{1.3}$$

where $\text{RMSE}(\boldsymbol{x}, \boldsymbol{y})$ refers to the root mean square error defined as $\text{RMSE} = \sqrt{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(\boldsymbol{x}_{ij} - \boldsymbol{y}_{ij})^2}$ for images $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{R}^{M\times N}$. As we can see, PSNR is closely related to the ℓ_2 norm distance between two images. The unit of PSNR is decibel (dB) and higher dB value indicates better image quality and lower RMSE. Even thouth PSNR is very simple and intuitive, higher PSNR does not indicate higher visual structural similarity. Hence, many researchers still make effort to find alternative and better IQA metrics.

SSIM [**ssim**]: Some researchers attempted to exploit the visual properties of the human visual system. One of the seminar work in this direction is the famous structural similarity index metric (SSIM), which is a full reference IQA metric. In SSIM, each image patch is separated into three different components indicating three core informative parts of the original patch. The three components are luminance (mean value of the pixels in the patch), contrast (the standard deviation of the patch), and structure (the mean subtracted patch). The major advantage of the SSIM is that it takes into account the fact that the human visual system is very sensitive tothe relative changes in luminance, rather than the absolute changes in luminance. The value range of the SSIM is between 0 and 1, where higher value indicate higher similarity (SSIM of 1 indicates that the two images are exactly the same). Note that it does not indicate that higher SSIM indicate better image quality, since the reference image is not the image of the best quality.

Other IQA Metrics: Besides of the PSNR and SSIM, there are many other IQA metrics for full reference and no reference IQA. Some examples include the MS-SSIM [**msssim**], which is a multi-scale extension of the original SSIM. Some examples in no reference IQA include BLIINDS [**bliinds**] and BIQI [**biqi**]. These IQA metrics capture the deviations from the expected statistics of the natural images. For example, BLIINDS measures the deviations from the expected histogram of certain features in DCT domain, while BIQI measures deviations from the expected distribution of wavelet coefficients in a multi-scale decomposition.

I have to mention that no IQA metric is perfect or best for image denoising task, both in full reference and no reference cases. The de facto standard metrics in image restoration community are PSNR and SSIM. In order to avoid these two metrics generate bad results, it is essential to demonstrate the image quality in the thesis for human subjective evaluation.

1.4 Literature Review

In this chapter, I will review the methods related to denoising in literature during the past decades. These methods can be divided into three categories. Firstly, I will review the denoising methods designed for additive white Gaussian noise (AWGN) since AWGN is the mostly studied noise distribution inthe literature. Though these methods are proposed to deal with the AWGN noise, the idea can be applied to the other image denoising tasks such as real color image denoising. Secondly, I will review the existing methods proposed for processing real noisy images. Though the methods in this domain is not that versartile than those methods for the AWGN noise, the real noisy image denoising is the current mainstream for the image denoising community. Due to the noise is not known beforehand, noise estimation should be performed for the real noisy image denoising task. Finally, I will also review the image noise estimation methods in the literature.

However, noise in real images is more complex than simple Gaussian or mixed Gaussian and impulse distribution. Besides, noise is usually unknown for existing methods. This makes image denoising still a challenging problem.

1.5 Synthetic Grayscale Image Denoising

As a classical problem in low level vision, image denoising has been extensively studied in the past decades, yet it is still an active topic for the reason that it provides an ideal test bed for image modeling techniques. In general, image denoising aims to recover the clean image $\mathbf{x}$ from its noisy observation $\mathbf{y} = \mathbf{x} + \mathbf{n}$, where $\mathbf{n}$ is the additive noise which is often assumed to be additive white Gaussian noise (AWGN). Other types of noise, e.g., Poisson noise and salt-and-pepper noise, are also been studied in the literature. The Poisson noise can be transformed into the additive noise after some transformation. Besides, the salt-and-pepper noise is naturally additive noise and can be formulated into the model mentioned above.

A variety of image denoising methods have been developed in past decades, including diffusion based methods [**PeronaMalik1990**], total variation based methods [**rudin1992nonlinear**; **osher2005iterative**], filtering based methods [**Tomasi1998**], wavelet/curvelet based methods [**softthresholding**; **bayesshrink**; **curvelet**], nonlocal self-similarity based methods [**nlm**; **bm3d**], sparse representation based methods [**ksvd**; **lssc**; **ncsr**], and low rank based methods [**nnm**; **wnnm**], etc. Recently, some discriminative denoising methods have also been developed by learning discriminative priors from pairs of clean and noisy images [**mlp**; **csf**; **tnrd**].

Image modeling plays a central role in image denoising. By modeling the wavelet transform coefficients as Laplacian distributions, many wavelet shrinkage based denoising methods such as the classical soft-thresholding [**softthresholding**] have been proposed. Chang et al. modeled the wavelet transform coefficients as generalized Gaussian distribution, and proposed the BayesShrink [**bayesshrink**] algorithm. By considering the correlation of wavelet coefficients across scales, Portilla et al. [**blsgsm**] proposed to use Gaussian Scale Mixtures for image modeling and achieved promising denoising performance. It is widely accepted that natural image gradients exhibit heavy-tailed distributions [**weiss**], and the total variation (TV) based methods [**rudin1992nonlinear**; **osher2005iterative**] actually assume Laplacian distributions of image gradients for denoising. The Fields of Experts (FoE) [**foe**] proposed by Roth and Black models the filtering responses with Student's t-distribution to learn filters through Markov Random Field (MRF) [**Bishop**]. Recently, Schmidt and Roth proposed the cascade of shrinkage fields (CSF) to perform denoising efficiently [**csf**].

Instead of modeling the image statistics in some transformed domain (e.g., gradient domain, wavelet domain or filtering response domain), another popular approach is to model the image priors on patches. One representative is the sparse representation based scheme which encodes an image patch as a linear combination of a few atoms selected from a dictionary [**olshausen1996emergence**; **olshausen1997sparse**; **ksvd**]. The dictionary can be chosen from the off-the-shelf dictionaries (e.g., wavelets and curvelets), or it can be learned from natural image patches. The seminal work of K-SVD [**ksvdtsp**; **ksvd**] has demonstrated promising denoising performance by dictionary learning, which has yet been extended and successfully used in various image processing and computer vision applications [**srcolor**; **srcvpr**; **lcksvd**]. By viewing image patches as samples of a multivariate variable vector and considering that natural images are non-Gaussian, Zoran and Weiss [**epll**; **gmmnips**] and Yu et al. [**ple**] used Gaussian Mixture Model (GMM) to model image patches, and achieved state-of-the-art denoising and image restoration results, respectively.

When the input is a noisy RGB color image, there are mainly three strategies for color image denoising. (1) The first strategy is to apply the grayscale image denoising algorithm to each channel. However, such a straightforward solution will not exploit the spectral correlation among RGB channels, and the denoising performance may not be very satisfying. (2) The second strategy is to transform the RGB image into a less correlated color space, such as YCbCr, and perform denoising in each channel of the transformed space [**foe**; **cbm3d**]. One representative work along this line is the CBM3D algorithm [**cbm3d**]. However, the color transform will complicate the noise distribution, and the correlation among color channels is not fully exploited. (3) The third strategy is to perform joint denoising on the RGB channels simultaneously for better use of the spectral correlation. For example, the patches from RGB channels are concatenated as a long vector for processing [**mairal2008sparse**; **Zhu_2016_CVPR**].

Though joint denoising of RGB channels is a more promising way for color image denoising, it is not a trivial extension from single channel (grayscale image) to multiple channels (color image). The noise in standard RGB (sRGB) space can be approximately modeled as AWGN, but it has different variances for different channels [**Liu2008**; **Leungtip**; **crosschannel2016**] due to the sensor characteristics and on-board processing steps in digital camera pipelines [**crosschannel2016**; **karaimer_brown_ECCV_2016**]. This makes

the real color image denoising problem much more complex. If the three channels are treated equally in the joint denoising process, false colors or artifacts can be generated [**mairal2008sparse**]. How to account for the different noise characteristics in color channels, and how to effectively exploit the within and cross channel correlation are the key issues for designing a good color image denoising method.

1.6 Realistic Color Image Denoising

During the last decade, a few methods have been proposed for real color image denoising. To the best of our knowledge, the study of real color image denoising can be traced back to the BLS-GSM model [**blsgsm**]. In [**blsgsm**], Portilla et al. proposed to use scale mixture of Gaussian in overcomplete oriented pyramids to estimate the latent clean images. In [**fullyblind**], Portilla proposed to use a correlated Gaussian model for noise estimation of each wavelet subband. Based on the robust statistics theory [**huber2011robust**], Rabie modeled the noisy pixels as outliers, which could be removed via Lorentzian robust estimator [**rabie2005robust**]. The CBM3D method [**cbm3d**] is a representative color image denoising method, which first transforms the RGB image into a luminance-chrominance space (e.g., YCbCr) and then applies the benchmark BM3D method [**bm3d**] to each channel separately. The non-local similar patches are grouped by the luminance channel. In [**Liu2008**], Liu et al. proposed the "Noise Level Function" to estimate the noise for each channel in natural images, and then use Gaussian conditional random field to obtain the latent clean image [**Liu2008**]. However, processing each channel separately would often achieve inferior performance to processing the color channels jointly [**mairal2008sparse**]. Later, Lebrun el al. proposed a multiscale denoising algorithm called 'Noise Clinic' [**noiseclinic**] for blind image denoising task. This method generalizes the NL-Bayes [**nlbayes**] to deal with signal and frequency dependent noise. Therefore, the methods [**noiseclinic**; **ncwebsite**; **Zhu_2016_CVPR**] perform real color image denoising by concatenating the patches of RGB channels into a long vector. However, the concatenation treats each channel equally and ignores the different noise statistics among these channels. The method in [**crosschannel2016**] models the cross-channel noise in real noisy images as multivariate Gaussian and the noise is removed by the Bayesian non-local means filter [**kervrann2007bayesian**]. The commercial software Neat Image [**neatimage**] estimates the noise parameters from a flat region

of the given noisy image and filters the noise accordingly. The methods in [**crosschannel2016**; **neatimage**] ignore the non-local self-similarity of natural images [**bm3d**; **wnnm**].

Despite the success of these methods, they have many limitations. On one hand, as suggested in [**Liu2008**; **noiseclinic**], Gaussian noise, assumed by [**fullyblind**; **rabie2005robust**; **Liu2008**], may be inflexible for more complex noise in real images. Hence, better approximation to the noise could bring better image denoising performance [**Liu2008**; **noiseclinic**]. Based on these observations, it is still needed to design an robust and effective model for blind image denoising. Few assumption and no parameter tuning would bring extra points.

1.7 Contribution

This thesis is mainly consisted by the several work I have done during my PhD study, during which I focus on designing new and better image denoising algorithms.

Firstly, I propose a method for denoising synthetic AWGN noise, from which we can study the performance of the nonlocal self-similarity priors of natural images. In fact, we propose to learn the external NSS priors and apply the learned model on denoising AWGN noise. The proposed method achieves state-of-the-art performance on AWGN denoising on both effectiveness and efficiency.

Basing on the success on the synthetic noise removal, I propose to exploit the power of the NSS priors in natural images to deal with the complex realistic noise in real-world noisy images. Specifically, we propose three methods exploiting the NSS priors of natural images for real noisy image denoising, which can be introduced as follows.

In the first method, I propose to learn the NSS prior from the external natural images, and then apply the learned external prior to guide the learning of the internal NSS prior of the input real noisy image. The experiments on two commonly used datasets and a new one we constructed to implement the shortage of existing datasets, demonstrate that the proposed method can achieve better performance than existing color image denoising methods such as CBM3D [**cbm3d**], the state-of-the-art Gaussian noise removal methods

[**bm3d**; **mlp**; **csr**], and the real noisy image denoising methdos [] including a commercial software Neat Image [**neatimage**], which is embeded into the famous PhotoShop CS for image processing tasks.

In the second method, I propose to employ the low rank model describe fully the the internal NSS prior, basing on the observed fact that the similar image patches can be contanated as a matrix of low rank. Different from the previous work, I extend the WNNM model and apply it to multi-channel version to make it feasible for color image denoising.

In the third method, is to use the sparse coding based method with additional weighting scheme to regard the local noise in real noisy images as a Gaussian and the prior is used to deal with the real noisy image.

Finally, to make my thesis more comprehensive, I construct a large benchmark of real noisy images captured by different types of famous commercial cameras, on which I also evaluate the image denoising methods mentioned above and the proposed methods in this thesis.

The structure of this thesis is organized as follows: in the chapter 2, we review the literatures in the image denoising area; in the chapter 3, we introduce the fully external method; in the chapter 4, we introduce the external prior guided internal method; in the chapter 5, we introduce the internal method based on low ran model; in the chapter 6, we introduce the internal method based on sparse coding model; in the chapter 7, we introduce the real noisy image dataset we construct, and finaly evaluate the proposed methods with the compared competing methods, both for synthetic AWGN or Poisson noise and real noise, including the commercial software designed especially for real noise.

1.8 Thesis Structure

Chapter 2: External Nonlocal Self-Similarity Prior Learning for Synthetic Gaussian Noise Removal

In this chapter, I will introduce our work on external nonlocal self-similarity (NSS) prior learning for synthetic Gaussian noise removal. As far as we know, this work is the first to learn the NSS priors of natural clean images, while

previous work only utilize the NSS priors of input noisy image for online denoising. The advantages of this offline learning is that it can preserve the details of natural images while being much faster then most online denoising methods.

Chapter 3: External Prior Guided Internal Prior Learning for Real Noisy Image Denoising

In this chapter, I will introduce our work on external prior guided internal prior learning method for real noisy image denoising. This work can maintain the advantages of both sides: from the external perspective, the method can preserve the structures of natural images better than the internal methods, while from the perspective of internal method, the proposed method can recover the details of the input noisy image better than the external methods.

Chapter 4: Multi-channel Weighted Nuclear Norm Minimization for Real Color Image Denoising

In this chapter, we introduce a multi-channel weighted nuclear norm minimization (MC-WNNM) method. This method regards different channels in RGB images differently to adaptively process the real color noisy images. Besides, this work also propose a new strategy for color image denoising. Experiments demonstrate that the proposed method can achieve better performance on real color image denoising than existing state-of-the-art methods, including some commercial software.

Chapter 5: A Triple Weighted Sparse Coding Scheme for Realistic Noisy Image Denoising

In this chapter, I introduce a novel sparse coding based method for real color image denoising. In this method, I regard the noise in each of the local region in the real noisy image as a Gaussian, and propose a triplely weighted scheme to deal with the complex realistic noise in real color noisy images. Experiments show that the proposed method performs better and faster than the nuclear norm based method mentioned in previous chapter.

Chapter 6: A Benchmark on Real Color Noisy Image, with Comprehensive Evaluation of State-of-the-art

To fully boost the research of real color noisy image denoising, we construct a large benchmark on real color noisy images. This dataset is collected from several representative cameras with comprehensive settings on contents, lighting, ISO, shutter, and aperture, etc. Based on this newly established dataset, we fully evaluated existing denoising methods, including the methods designed for synthetic Gaussian noise and the methods designed especially for real color noise. We believe that this new dataset will largely boost the research of the image denoising especially the realistic image denoising problems.

2 External Non-local Self-Similarity Prior for Additive White Gaussian Noise

2.1 Introduction

As a classical problem in low level vision, image denoising has been extensively studied, yet it is still an active topic for that it provides an ideal test bed for image modeling techniques. In general, image denoising aims to recover the clean image $\boldsymbol{x}$ from its noisy observation $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{v}$, where $\boldsymbol{v}$ is assumed to be additive white Gaussian noise. A variety of image denoising methods have been developed in past decades, including filtering based methods [**Tomasi1998**], diffusion based methods [**PeronaMalik1990**], total variation based methods [**rudin1992nonlinear**; **osher2005iterative**], wavelet/curvelet based methods [**softthresholding**; **bayesshrink**; **curvelet**], sparse representation based methods [**ksvd**; **lssc**; **ncsr**], nonlocal self-similarity based methods [**nlm**; **bm3d**; **nnm**; **wnnm**], etc.

Image modeling plays a central role in image denoising. By modeling the wavelet transform coefficients as Laplacian distributions, many wavelet shrinkage based denoising methods such as the classical soft-thresholding [**softthresholding**] have been proposed. Chang et al. modeled the wavelet transform coefficients as generalized Gaussian distribution, and proposed the BayesShrink [**bayesshrink**] algorithm. By considering the correlation of wavelet coefficients across scales, Portilla et al. [**blsgsm**] proposed to use Gaussian Scale Mixtures for image modeling and achieved promising denoising performance. It is widely accepted that natural image gradients exhibit heavy-tailed distributions [**weiss**], and the total variation (TV) based methods [**rudin1992nonlinear**; **osher2005iterative**] actually assume Laplacian distributions of image gradients for denoising. The Fields of Experts (FoE) [**foe**] proposed by Roth and Black models the filtering responses with Student's t-distribution to learn filters through Markov Random Field (MRF) [**Bishop**]. Recently, Schmidt and Roth proposed the cascade of shrinkage fields (CSF) to perform denoising efficiently [**csf**].

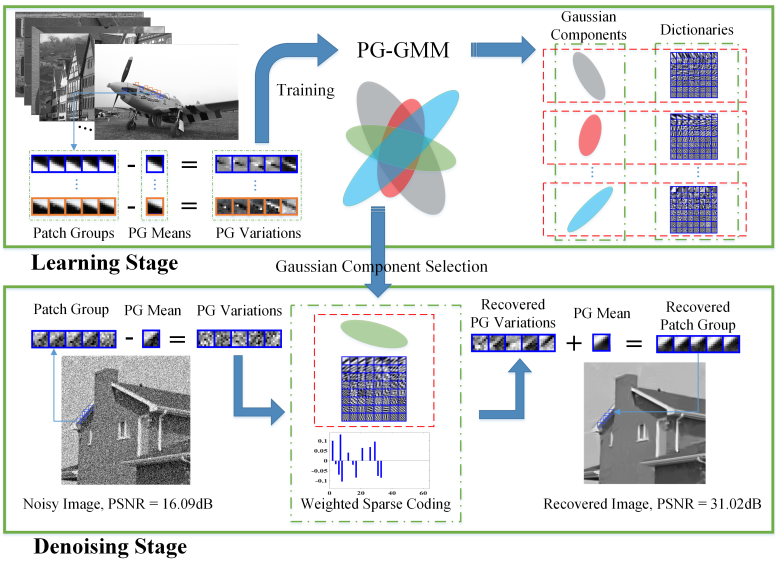


Fig. 2.1: Flowchart of the proposed patch group based prior learning and image denoising framework.

Instead of modeling the image statistics in some transformed domain (e.g., gradient domain, wavelet domain or filtering response domain), another popular approach is to model the image priors on patches. One representative is the sparse representation based scheme which encodes an image patch as a linear combination of a few atoms selected from a dictionary [**olshausen1996emergence**; **olshausen1997sparse**; **ksvd**]. The dictionary can be chosen from the off-the-shelf dictionaries (e.g., wavelets and curvelets), or it can be learned from natural image patches. The seminal work of K-SVD [**ksvdtsp**; **ksvd**] has demonstrated promising denoising performance by dictionary learning, which has yet been extended and successfully used in various image processing and computer vision applications [**srcolor**; **srcvpr**; **lcksvd**]. By viewing image patches as samples of a multivariate variable vector and considering that natural images are non-Gaussian, Zoran and Weiss [**epll**; **gmmnips**] and Yu et al. [**ple**] used Gaussian Mixture Model (GMM) to model image patches, and achieved state-of-the-art denoising and image restoration results, respectively.

Natural images often have many repetitive local patterns, and a local patch can have many similar patches to it across the whole image. The so-called nonlocal self-similarity (NSS) prior is among the most successful priors for image restoration. The nonlocal means [**nlm**] and nonlocal regularization [**nlr**] methods improve much the image denoising performance over the conventional local self-similarity based methods. Dabov et al. [**bm3d**] constructed 3D cubes of nonlocal similar patches and conducted collaborative

filtering in the sparse 3D transform domain. The so-called BM3D algorithm has become a benchmark in image denoising. Mairal et al. [**lssc**] proposed the LSSC algorithm to exploit NSS via group sparse coding. The NSP [**nsp**] method fits the singular values of NSS patch matrix by Laplacian distribution. Dong et al. [**ncsr**] unified NSS and local sparse coding into the so-called NCSR framework, which shows powerful image restoration capability. By assuming that the matrix of nonlocal similar patches has a low rank structure, the low-rank minimization based methods [**nnm**; **wnnm**] have also achieved very competitive denoising results.

Though NSS has demonstrated its great success in image denoising, in most existing methods only the NSS of noisy input image is used for denoising. For example, in BM3D [**bm3d**] the nonlocal similar patches of a noisy image are collected as a cube for collaborative filtering. In NCSR [**ncsr**], the nonlocal means are subtracted in the sparse domain to regularize the sparse coding of noisy patches. In WNNM [**wnnm**], the low-rank regularization is enforced to recover the latent structure of the matrix of noisy patches. We argue that, however, such utilizations of NSS are not effective enough because they neglect the NSS of clean natural images, which can be pre-learned for use in the denoising stage. To the best of our knowledge, unfortunately, so far there is not an explicit NSS prior model learned from natural images for image restoration.

With the above considerations, in this work we propose to learn explicit NSS models from natural images, and apply the learned prior models to noisy images for high performance denoising. The flowchart of the proposed method is illustrated in Fig. 4.1. In the learning stage, we extract millions of patch groups (PG) from a set of clean natural images. A PG is formed by grouping the similar patches to a local patch in a large enough neighborhood. A PG based GMM (PG-GMM) learning algorithm is developed to learn the NSS prior for the PGs. In the denoising stage, the learned PG-GMM will provide dictionaries as well as regularization parameters, and a simple weighted sparse coding model is developed for image denoising. Our extensive experiments validated that the proposed PG prior based denoising method outperforms many state-of-the-art algorithms quantitatively (in PSNR) while being much more efficient. More importantly, it delivers the best qualitative denoising results with finer details and less artifacts, owe to the NSS prior learned from clean natural images.

2.2 Patch Group Based Prior Modeling of Nonlocal Self-Similarity

Image nonlocal self-similarity (NSS) has been widely adopted in patch based image denoising and other image restoration tasks [**nlm**; **bm3d**; **lssc**; **ncsr**; **wnnm**]. Despite the great success of NSS in image restoration, most of the existing works exploit the NSS only from the degraded image. Usually, for a given patch in the degraded image, its nonlocal similar patches are collected, and then the nonlocal means [**nlm**], or 3D transforms [**bm3d**], or some regularization terms [**lssc**; **ncsr**; **wnnm**; **srcolor**] can be introduced for image restoration. However, how to learn the NSS prior from clean natural images and apply it to image restoration is still an open problem. In this work, we make the first attempt on this problem, and develop a patch group (PG) based NSS prior learning scheme.

2.2.1 Patch Group and Group Mean Subtraction

For each local patch (size: $p \times p$) of a given clean image, we can find the first M most similar nonlocal patches to it across the whole image. In practice, this can be done by Euclidean distance based block matching in a large enough local window of size $W \times W$. A PG is formed by grouping the M similar patches, denoted by $\{\boldsymbol{x}_m\}_{m=1}^{M}$, where $\boldsymbol{x}_m \in \mathbb{R}^{p^2 \times 1}$ is a patch vector. The mean vector of this PG is $\boldsymbol{\mu} = \frac{1}{M}\sum_{m=1}^{M} \boldsymbol{x}_m$, and $\overline{\boldsymbol{x}}_m = \boldsymbol{x}_m - \boldsymbol{\mu}$ is the group mean subtracted patch vector. We call

$$\overline{\boldsymbol{X}} \triangleq \{\overline{\boldsymbol{x}}_m\}, m = 1, ..., M \tag{2.1}$$

the group mean subtracted PG, and it will be used to learn the NSS prior in our work.

In Fig. 4.2, we show two different PGs, their group means, and the PGs after mean subtraction. One can see that before mean subtraction, the two PGs have very different local structures. After mean subtraction, the two PGs will have very similar variations. This greatly facilitates the prior learning because the possible number of patterns is reduced, while the training samples of each pattern are increased. We will discuss further the benefits of mean subtraction and the associated prior model learning in Section 2.4.

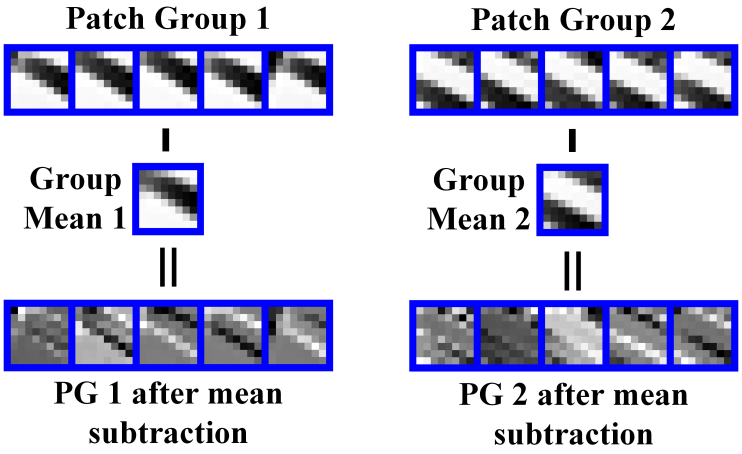


Fig. 2.2: Different patch groups (PG) share similar PG variations.

2.2.2 PG-GMM Learning

From a given set of natural images, we can extract N PGs, and we denote one PG as

$$\overline{\boldsymbol{X}}_n \triangleq \{\overline{\boldsymbol{x}}_{n,m}\}_{m=1}^{M}, n = 1, ..., N. \tag{2.2}$$

The PGs $\{\overline{\boldsymbol{X}}_n\}$ contain a rich amount of NSS information of natural images, and the problem turns to how to learn explicit prior models from $\{\overline{\boldsymbol{X}}_n\}$. Considering that Gaussian Mixture Model (GMM) has been successfully used to model the image patch priors in EPLL [**epll**] and PLE [**ple**], we propose to extend patch based GMM to patch group based GMM (PG-GMM) for NSS prior learning.

With PG-GMM, we aim to learn a set of K Gaussians $\{\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}$ from N training PGs $\{\overline{\boldsymbol{X}}_n\}$, while requiring that all the M patches $\{\overline{\boldsymbol{x}}_{n,m}\}$ in PG $\overline{\boldsymbol{X}}_n$ belong to the same Gaussian component and assume that the patches in the PG are independently sampled. Note that such an assumption is commonly used in patch based image modeling [**ksvd**; **lssc**]. Then, the likelihood of $\{\overline{\boldsymbol{X}}_n\}$ can be calculated as

$$P(\overline{\boldsymbol{X}}_n) = \sum_{k=1}^{K} \pi_k \prod_{m=1}^{M} \mathcal{N}(\overline{\boldsymbol{x}}_{n,m} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{2.3}$$

By assuming that all the PGs are independently sampled, the overall objective likelihood function is $\mathcal{L} = \prod_{n=1}^{N} P(\overline{\boldsymbol{X}}_n)$. Taking the log of it, we maximize the following objective function for PG-GMM learning

$$\ln \mathcal{L} = \sum_{n=1}^{N} \ln(\sum_{k=1}^{K} \pi_k \prod_{m=1}^{M} \mathcal{N}(\overline{\boldsymbol{x}}_{n,m}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)). \tag{2.4}$$

As in GMM learning [**Bishop**], we introduce hidden variables $\{\Delta_{nk}|n = 1, ..., N; k = 1, ..., K\}$ to optimize (3.4). If PG $\overline{\boldsymbol{X}}_n$ belongs to the kth component, $\Delta_{nk} = 1$; and $\Delta_{nk} = 0$ otherwise. Then the EM algorithm [**em**] can be used to optimize (3.4) via two alternative steps. In the E-Step, by the Bayes' formula, the expected value of Δ_{nk} is

$$\gamma_{nk} = \frac{\pi_k \prod_{m=1}^{M} \mathcal{N}(\overline{\boldsymbol{x}}_{n,m}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{l=1}^{K} \pi_l \prod_{m=1}^{M} \mathcal{N}(\overline{\boldsymbol{x}}_{n,m}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}. \tag{2.5}$$

In the M-step, since for each PG $\overline{\boldsymbol{X}}_n$, $\sum_{m=1}^{M} \overline{\boldsymbol{x}}_{n,m} = \mathbf{0}$, we have

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \gamma_{nk} \sum_{m=1}^{M} \overline{\boldsymbol{x}}_{n,m}}{\sum_{n=1}^{N} \gamma_{nk}} = \mathbf{0}, \tag{2.6}$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_{n=1}^{N} \gamma_{nk} \sum_{m=1}^{M} \overline{\boldsymbol{x}}_{n,m} \overline{\boldsymbol{x}}_{n,m}^T}{\sum_{n=1}^{N} \gamma_{nk}}. \tag{2.7}$$

The calculations of $\pi_k = \frac{1}{N} \sum_{n=1}^{N} \gamma_{nk}$ are similar to [**Bishop**].

By alternating between the E-step and the M-step, the model parameters will be updated iteratively, and the update in each iteration can guarantee to increase the value of the log-likelihood function (3.5), and the EM algorithm will converge [**Bishop**; **emconvergence**]. Fig. 5.3 shows the convergence curve of the proposed PG-GMM algorithm by using the Kodak PhotoCD Dataset (`http://r0k.us/graphics/kodak/`) for training.

2.2.3 Complexity Analysis

In the training stage, there are N PGs, each of which has M patches, and hence we have $N \times M$ patches. In the M-step, we only need to calculate the covariance matrices since the mean of each Gaussian component is zero. The cost of this step is $O(p^4MN)$. In the E-step, the cost is $O(p^6MN)$. Suppose

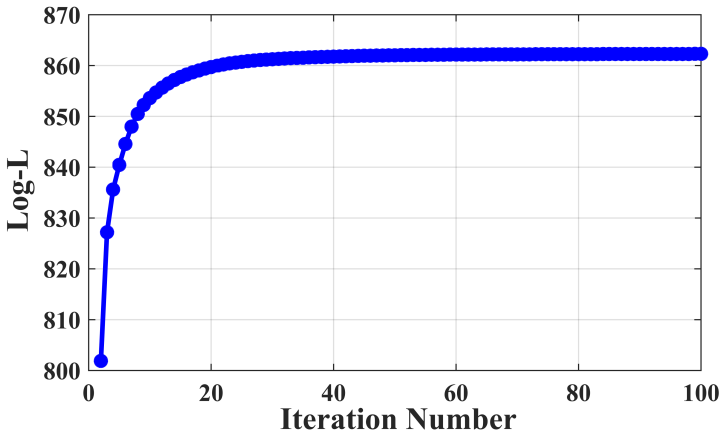


Fig. 2.3: The convergence curve of log-likelihood in PG-GMM training on the Kodak PhotoCD Dataset.

that the number of iterations is T, the overall complexity of PG-GMM training is $O(p^6MNT)$.

2.2.4 Discussions

GMM has been used for patch based image prior learning and achieved promising results, e.g., EPLL [**epll**] and PLE [**ple**]. In this paper, we extend the patch based image prior learning to PG based prior learning to model the NSS information. The developed PG-GMM method has some important advantages over the patch based GMM method.

First, in patch based GMM, the mean value of each patch is subtracted before learning the Gaussian components. This is to remove the DC (direct current) of each patch but will not change the essential structure of a patch. However, in PG-GMM the mean vector of all patches in a group is calculated and subtracted from each patch, and hence the structure of each patch is changed. As a result, many patches which originally have different local patterns may become similar after group mean subtraction (please refer to Fig. 4.2 for an example). This makes the PG-GMM learning process easier and more stable.

Second, as can be seen in Eq. (3.9), the mean vector of each Gaussian component in PG-GMM is naturally a zero vector. This implies that we only need to learn the covariance matrix of each component without considering its mean. However, in patch based GMM [**epll**], the mean vectors of Gaussians can only be forced to zero and there is no theoretical guarantee for this.

Third, due to reduction of possible patterns in PG-GMM and the reduced number of variables to learn, we do not need to set a large number of Gaussian components in PG-GMM learning. For example, in EPLL [**epll**], 200 Gaussian components are learned to achieve competing denoising performance with BM3D [**bm3d**], while in PG-GMM learning only 32 Gaussian components are enough to outperform BM3D (please refer to the experimental section for details).

2.3 Image Denoising by Patch Group Priors

2.3.1 Denoising Model

Given a noisy image $\boldsymbol{y}$, like in the PG-GMM learning stage, for each local patch we search for its similar patches in a window centered on it to form a PG, denoted by $\boldsymbol{Y} = \{\boldsymbol{y}_1, ..., \boldsymbol{y}_M\}$. Then the group mean of $\boldsymbol{Y}$, denoted by $\boldsymbol{\mu}_y$, is calculated and subtracted from each patch, leading to the mean subtracted PG $\overline{\boldsymbol{Y}}$. We can write $\overline{\boldsymbol{Y}}$ as $\overline{\boldsymbol{Y}} = \overline{\boldsymbol{X}} + \boldsymbol{V}$, where $\overline{\boldsymbol{X}}$ is the corresponding clean PG and $\boldsymbol{V}$ contains the corrupted noise. The problem then turns to how to recover $\overline{\boldsymbol{X}}$ from $\overline{\boldsymbol{Y}}$ by using the learned PG-GMM priors. Note that the mean $\boldsymbol{\mu}_y$ of $\overline{\boldsymbol{Y}}$ is very close to the mean of $\overline{\boldsymbol{X}}$ since the mean vector of noise $\boldsymbol{V}$ is nearly zero. $\boldsymbol{\mu}_y$ will be added back to the denoised PG to obtain the denoised image.

Gaussian Component Selection

For each $\overline{\boldsymbol{Y}}$, we select the most suitable Gaussian component to it from the trained PG-GMM. As in [**epll**], suppose that the variance of Gaussian white noise corrupted in the image is σ^2, the covariance matrix of the kth component will become $\boldsymbol{\Sigma}_k + \sigma^2\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. The selection can be done by checking the posterior probability that $\overline{\boldsymbol{Y}}$ belongs to the kth Gaussian component:

$$P(k|\overline{\boldsymbol{Y}}) = \frac{\prod_{m=1}^{M} \mathcal{N}(\overline{\boldsymbol{y}}_m|\boldsymbol{0}, \boldsymbol{\Sigma}_k + \sigma^2\boldsymbol{I})}{\sum_{l=1}^{K}\prod_{m=1}^{M} \mathcal{N}(\overline{\boldsymbol{y}}_m|\boldsymbol{0}, \boldsymbol{\Sigma}_l + \sigma^2\boldsymbol{I})}. \tag{2.8}$$

Taking log-likelihood of (3.8), we have

$$\ln P(k|\overline{\boldsymbol{Y}}) = \sum_{m=1}^{M} \ln \mathcal{N}(\overline{\boldsymbol{y}}_m|\boldsymbol{0}, \boldsymbol{\Sigma}_k + \sigma^2) - \ln C \tag{2.9}$$

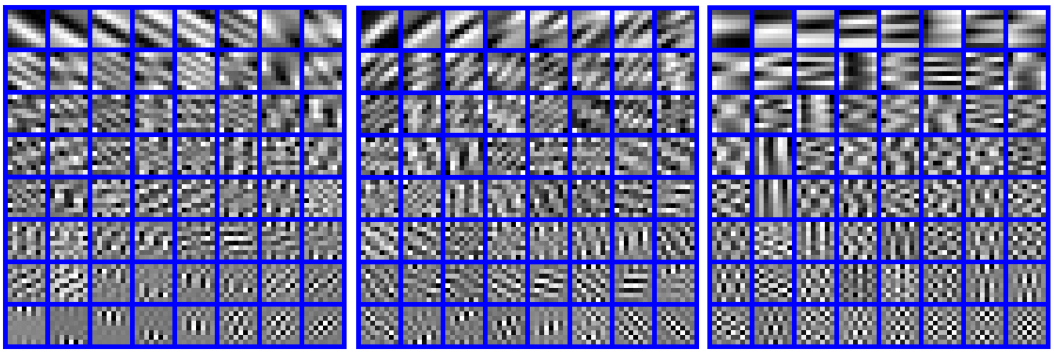


Fig. 2.4: Eigenvectors of 3 Gaussian components from the learned PG-GMM, sorted by the values of corresponding eigenvalues.

where C is the denominator in Eq. (3.9) and it is the same for all components. Finally, the component with the highest probability $\ln P(k|\overline{\boldsymbol{Y}})$ is selected to process $\overline{\boldsymbol{Y}}$.

Weighted Sparse Coding with Closed-Form Solution

Suppose that the kth Gaussian component is selected for PG $\overline{\boldsymbol{Y}}$. For notation simplicity, we remove the subscript k and denote by $\boldsymbol{\Sigma}$ the covariance matrix of this component. In PG-GMM, the PGs actually represent the variations of the similar patches in a group, and these variations are assigned to the same Gaussian distribution. By singular value decomposition (SVD), $\boldsymbol{\Sigma}$ can be factorized as

$$\boldsymbol{\Sigma} = \boldsymbol{D}\boldsymbol{\Lambda}\boldsymbol{D}^T, \tag{2.10}$$

where $\boldsymbol{D}$ is an orthonormal matrix composed by the eigenvectors of $\boldsymbol{\Sigma}$ and $\boldsymbol{\Lambda}$ is the diagonal matrix of eigenvalues. With PG-GMM, the eigenvectors in $\boldsymbol{D}$ capture the statistical structures of NSS variations in natural images, while the eigenvalues in $\boldsymbol{\Lambda}$ represent the significance of these eigenvectors. Fig. 4 shows the eigenvectors for 3 Gaussian components. It can be seen that these eigenvectors encode the possible variations of the PGs. For one Gaussian component, the first eigenvector represents its largest variation, while the last eigenvector represents its smallest variation. For different Gaussian components, we can see that their eigenvectors (with the same index) are very different. Hence, $\boldsymbol{D}$ can be used to represent the structural variations of the PGs in that component.

For each patch $\overline{\boldsymbol{y}}_m$ in the PG $\overline{\boldsymbol{Y}}$, we propose to use $\boldsymbol{D}$ as the dictionary to sparsely encode $\overline{\boldsymbol{y}}_m$ as $\overline{\boldsymbol{y}}_m = \boldsymbol{D}\boldsymbol{\alpha} + \boldsymbol{v}$, where $\boldsymbol{\alpha}$ is the vector of sparse coding

coefficients and $\boldsymbol{v}$ is the corrupted noise. Meanwhile, we propose to introduce a weighting vector $\boldsymbol{w}$ to weight the coding vector $\boldsymbol{\alpha}$ (we will see in (8.3) that $\boldsymbol{w}$ is related to the eigenvalues in $\boldsymbol{\Lambda}$), resulting in the following simple but highly effective weighted sparse coding model:

$$\min_{\boldsymbol{\alpha}} \|\overline{\boldsymbol{y}}_m - \boldsymbol{D}\boldsymbol{\alpha}\|_2^2 + \|\boldsymbol{w}^T\boldsymbol{\alpha}\|_1. \tag{2.11}$$

From the viewpoint of Maximum A-Posterior (MAP) estimation, the optimal solution of (3.11) is $\hat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha}} \ln P(\boldsymbol{\alpha}|\overline{\boldsymbol{y}}_m)$. By Bayes' formula, it is equivalent to

$$\hat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha}}\{\ln P(\overline{\boldsymbol{y}}_m|\boldsymbol{\alpha}) + \ln P(\boldsymbol{\alpha})\}. \tag{2.12}$$

The log-likelihood term $\ln P(\overline{\boldsymbol{y}}_m|\boldsymbol{\alpha})$ is characterized by the statistics of noise $\boldsymbol{v}$, which is assumed to be white Gaussian with standard deviation σ. Hence, we have

$$P(\overline{\boldsymbol{y}}_m|\boldsymbol{\alpha}) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}\|\overline{\boldsymbol{y}}_m - \boldsymbol{D}\boldsymbol{\alpha}\|_2^2). \tag{2.13}$$

We assume that the sparse coding coefficients in $\boldsymbol{\alpha}$ follow i.i.d. Laplacian distribution. More specifically, for entry $\boldsymbol{\alpha}_i$, which is the coding coefficient of patch $\overline{\boldsymbol{y}}_m$ over the ith eigenvector in $\boldsymbol{D}$, we assume that it follows distribution $\frac{c}{\sqrt{2}\boldsymbol{\lambda}_i}\exp(-c\sqrt{2}|\boldsymbol{\alpha}_i|/\boldsymbol{\lambda}_i)$, where $\boldsymbol{\lambda}_i = \boldsymbol{\Lambda}_i^{1/2}$ and c is a constant. Note that we adjust the scale factor of the distribution by (square root of) the ith eigenvalue $\boldsymbol{\Lambda}_i$. This is because the larger the eigenvalue $\boldsymbol{\Lambda}_i$ is, the more important the ith eigenvector in $\boldsymbol{D}$ is, and hence the distribution of the coding coefficients over this eigenvector should have a longer tail (i.e., less sparse). Finally, we have

$$P(\boldsymbol{\alpha}) = \prod_{i=1}^{p^2}\frac{c}{\sqrt{2}\boldsymbol{\lambda}_i}\exp(-\frac{c\sqrt{2}|\boldsymbol{\alpha}_i|}{\boldsymbol{\lambda}_i}). \tag{2.14}$$

Putting (3.13) and (8.1) into (3.12), we have

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \|\overline{\boldsymbol{y}}_m - \boldsymbol{D}\boldsymbol{\alpha}\|_2^2 + \sum_{i=1}^{p^2}\frac{c*2\sqrt{2}\sigma^2}{\boldsymbol{\lambda}_i}|\boldsymbol{\alpha}_i|. \tag{2.15}$$

By comparing (8.2) with (3.11), we can see that the ith entry of the weighting vector $\boldsymbol{w}$ should be

$$\boldsymbol{w}_i = c*2\sqrt{2}\sigma^2/(\boldsymbol{\lambda}_i + \varepsilon), \tag{2.16}$$

where ε is a small positive number to avoid dividing by zero.

With $\boldsymbol{w}$ determined by (8.3), let's see what the solution of (3.11) should be. Since the dictionary $\boldsymbol{D}$ is orthonormal, it is not difficult to find out that

(3.11) has a closed-form solution (detailed derivation can be found in the supplementary material):

$$\hat{\boldsymbol{\alpha}} = \text{sgn}(\boldsymbol{D}^{\boldsymbol{T}}\overline{\boldsymbol{y}}_m) \odot \max(|\boldsymbol{D}^{\boldsymbol{T}}\overline{\boldsymbol{y}}_m| - \boldsymbol{w}/2, 0), \tag{2.17}$$

where sgn(•) is the sign function, $\odot$ means element-wise multiplication, and $|\boldsymbol{D}^{\boldsymbol{T}}\overline{\boldsymbol{y}}_m|$ is the absolute value of each entry of vector $|\boldsymbol{D}^{\boldsymbol{T}}\overline{\boldsymbol{y}}_m|$. The closed-form solution makes our weighted sparse coding process very efficient.

2.3.2 Denoising Algorithm

With the solution $\hat{\boldsymbol{\alpha}}$ in (8.4), the clean patch in a PG can be estimated as $\hat{\boldsymbol{x}}_m = \boldsymbol{D}\hat{\boldsymbol{\alpha}} + \boldsymbol{\mu}_y$. Then the clean image $\hat{\boldsymbol{x}}$ can be reconstructed by aggregating all the estimated PGs. In practice, we could perform the above denoising procedures for several iterations for better denoising outputs. In iteration t, we use the iterative regularization strategy [**osher2005iterative**] to add back to the recovered image $\hat{\boldsymbol{x}}^{(t-1)}$ some estimation residual in iteration $t-1$. The standard deviation of noise in iteration t is adjusted as $\sigma^{(t)} = \eta * \sqrt{\sigma^2 - \|\boldsymbol{y} - \boldsymbol{y}^{(t-1)}\|_2^2}$, where η is a constant. The proposed denoising algorithm is summarized in Algorithm 1 (Alg. 1).

In the proposed algorithm, there are N PGs in an image and M patches in each PG. Then the computational cost for Gaussian component selection is $O(p^6NMK)$. The cost for iterative regularization and noise estimation is negligible. The cost for closed-form weighted sparse coding is $O(p^4NM)$. Suppose that there are T iterations, the overall complexity of our denoising algorithm is $O(p^6NMKT)$.

2.4 Experiments

In this section, we perform image denoising experiments on 20 widely used natural images (shown in Fig. 3.5). More experiments on the Berkeley Segmentation Data Set [**bsds**] can be found in the supplementary file. As a common experimental setting in literature, additive white Gaussian noise with zero mean and standard deviation σ is added to the image to test the performance of competing denoising methods. We call our method *PG Prior based Denoising* (PGPD) in the following experiments. The Matlab

Alg. 1: Patch Group Prior based Denoising (PGPD)

Input: Noisy image $\boldsymbol{y}$, PG-GMM model

1. Initialization: $\hat{\boldsymbol{x}}^{(0)} = \boldsymbol{y}, \boldsymbol{y}^{(0)} = \boldsymbol{y}$;

for $t = 1 : IteNum$ **do**

2. Iterative Regularization:
$$\boldsymbol{y}^{(t)} = \hat{\boldsymbol{x}}^{(t-1)} + \delta(\boldsymbol{y} - \boldsymbol{y}^{(t-1)});$$
3. Estimate the standard deviation of noise;

for each PG $\boldsymbol{Y}$ **do**

4. Calculate group mean $\boldsymbol{\mu}_y$ and form PG $\overline{\boldsymbol{Y}}$;
5. Gaussian component selection via (3.9);
6. Denoising by Weighted Sparse Coding (8.2);
7. Recover each patch in this PG via $\hat{\boldsymbol{x}}_m = \boldsymbol{D}\hat{\boldsymbol{\alpha}} + \boldsymbol{\mu}_y$;

end for

8. Aggregate the recovered PGs to form the recovered image $\hat{\boldsymbol{x}}^{(t)}$;

end for

Output: The recovered image $\hat{\boldsymbol{x}}^{(IteNum)}$.

Fig. 2.5: The 20 widely used test images.

source code of our PGPD algorithm can be downloaded at `http://www4.comp.polyu.edu.hk/~cslzhang/code/PGPD.zip`.

2.4.1 Implementation Details

Our proposed PGPD method contains two stages, the prior learning stage and the denoising stage. In the PG-GMM learning stage, there are 4 parameters: p, M, W and K. The patch size ($p \times p$) is set as $p = 6$ for $0 < \sigma \leq 20$, $p = 7$ for $20 < \sigma \leq 30$, $p = 8$ for $30 < \sigma \leq 50$, and $p = 9$ for $50 < \sigma \leq 100$. The window size (W) for PG searching is set to $W = 31$. The number (M) of patches in a PG is set to $M = 10$. The number (K) of Gaussian components is set to $K = 64$ for $p = 6$ and $K = 32$ otherwise. We extracted about one million PGs from the Kodak PhotoCD Dataset to train the PG-GMM.

In the denoising stage, there are 3 parameters: c, δ, and η. In our implementation, (c, δ, η) are set to $(0.33, 0.10, 0.79)$, $(0.29, 0.09, 0.73)$, $(0.19, 0.08, 0.89)$, $(0.15, 0.07, 0.98)$, $(0.12, 0.06, 1.05)$, $(0.09, 0.05, 1.15)$, $(0.06, 0.05, 1.30)$ when $\sigma =$

Table 2.1: PSNR(dB) results of PPD and PGPD on the 20 natural images.

	$\sigma = 10$		$\sigma = 20$		$\sigma = 30$		$\sigma = 40$		$\sigma = 50$		$\sigma = 75$		$\sigma = 100$	
Images	**PPD**	**PGPD**	**PPD**	**PGPD**	**PPD**	**PGPD**	**PPD**	**PGPD**	**PPD**	**PGPD**	**PPD**	**PGPD**	**PPD**	**PGPD**
Airfield	31.02	31.18	27.97	28.19	26.33	26.46	25.20	25.30	24.33	24.44	22.69	22.90	21.54	21.82
Airplane	35.87	36.00	32.59	32.69	30.62	30.80	29.21	29.44	28.10	28.38	25.90	26.39	24.35	25.01
Baboon	30.46	30.55	26.54	26.67	24.54	24.63	23.23	23.39	22.30	22.47	20.71	21.09	19.99	20.38
Barbara	33.96	34.74	30.24	31.40	27.97	29.38	26.29	27.97	24.94	26.81	22.84	24.84	22.04	23.48
Boat	33.58	33.77	30.58	30.82	28.80	29.05	27.51	27.82	26.52	26.85	24.72	25.19	23.56	24.06
C. Man	33.91	34.14	30.12	30.35	28.25	28.53	27.05	27.33	26.13	26.46	24.36	24.64	22.80	23.23
Carhouse	34.35	34.47	30.61	30.73	28.62	28.80	27.29	27.51	26.27	26.53	24.44	24.85	23.21	23.67
Couple	33.78	34.03	30.47	30.71	28.54	28.84	27.16	27.53	26.07	26.50	24.22	24.70	23.12	23.55
Elaine	32.73	32.98	31.21	31.32	30.24	30.37	29.42	29.62	28.69	28.90	27.26	27.47	26.17	26.27
Hat	35.44	35.44	31.42	31.44	29.05	29.31	27.43	27.90	26.28	26.76	24.19	24.79	22.86	23.45
Hill	33.38	33.58	30.41	30.66	28.85	29.09	27.76	28.06	26.91	27.22	25.34	25.73	24.36	24.66
House	35.61	36.56	33.18	33.85	31.62	32.24	30.32	31.02	29.17	29.93	26.81	27.81	25.13	26.17
Lake	32.86	32.98	30.00	30.09	28.30	28.38	27.03	27.15	26.05	26.20	24.19	24.49	22.94	23.36
Leaves	33.87	34.45	29.84	30.46	27.51	27.99	25.88	26.29	24.56	25.03	21.94	22.61	19.77	20.95
Lena	35.59	35.81	32.75	32.94	30.98	31.27	29.67	30.10	28.61	29.11	26.68	27.40	25.41	26.09
Man	33.91	33.98	30.52	30.60	28.72	28.86	27.53	27.73	26.63	26.86	25.01	25.36	24.00	24.33
Monarch	34.27	34.53	30.40	30.68	28.27	28.49	26.81	27.02	25.66	26.00	23.51	24.00	21.89	22.56
Paint	34.16	34.31	30.52	30.62	28.39	28.42	26.88	26.94	25.70	25.82	23.50	23.89	22.05	22.65
Peppers	34.71	34.82	32.61	32.66	31.13	31.25	29.95	30.18	28.99	29.22	27.04	27.42	25.45	25.94
Zelda	35.50	35.51	32.21	32.21	30.35	30.43	29.07	29.23	28.06	28.24	26.37	26.56	25.28	25.41
Average	33.95	34.19	30.71	30.95	28.85	29.13	27.53	27.88	26.50	26.89	24.59	25.11	23.30	23.85

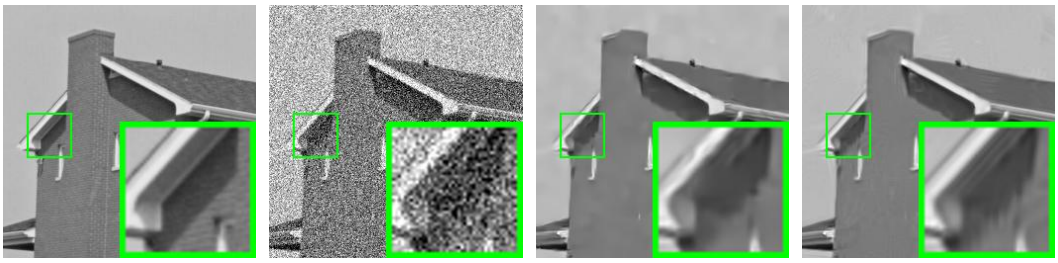
(a) Clean **(b)**Noisy: 18.59 **(c)**PPD: 28.85 **(d)**PGPD: 29.09

Fig. 2.6: Denoised images and PSNR (dB) results of *Hill* by the PPD method and PGPD method (the standard deviation of noise is $\sigma = 30$).

$10, 20, 30, 40, 50, 75, 100$, respectively. In addition, on all noise levels we stop Algorithm 1 in 4 iterations.

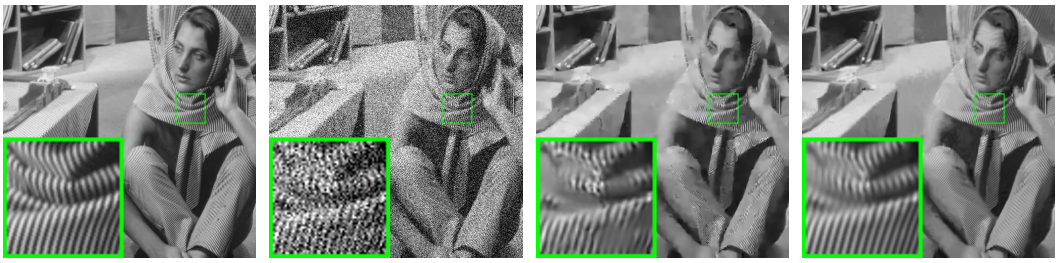
2.4.2 Comparison with Patch Prior based Denoising

In this section, we compare the PSNR and visual quality of denoised images by the *Patch Prior based Denoising* (PPD) method and the *Patch Group Prior based Denoising* (PGPD) method. As can be seen from Table 3.1 and Figures 4.1-4.4, PGPD is much better than PPD both quantitatively and qualitatively. This validates the effectiveness of our learned PG based NSS prior. In the following sections, we will omit the results of the PPD method.



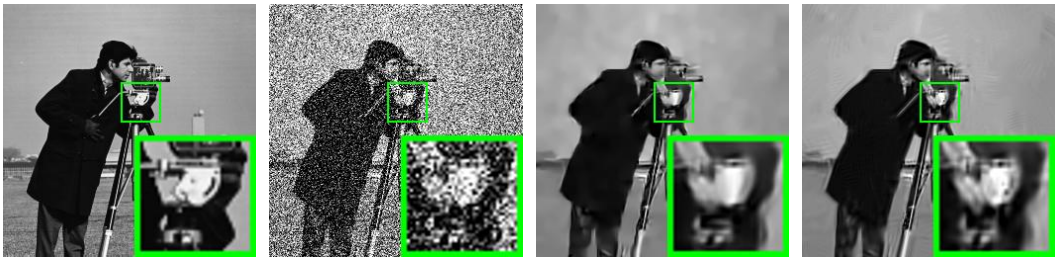
(a) Clean **(b)**Noisy: 16.06 **(c)**PPD: 30.32 **(d)**PGPD: 31.02

Fig. 2.7: Denoised images and PSNR (dB) results of *House* by the PPD method and PGPD method (the standard deviation of noise is $\sigma = 40$).



(a) Clean **(b)**Noisy: 14.16 **(c)**PPD: 24.94 **(d)**PGPD: 26.81

Fig. 2.8: Denoised images and PSNR (dB) results of *Barbara* by the PPD method and PGPD method (the standard deviation of noise is $\sigma = 50$).



(a) Clean **(b)**Noisy: 10.60 **(c)**PPD: 24.36 **(d)**PGPD:24.64

Fig. 2.9: Denoised images and PSNR (dB) results of *Cameraman* by the PPD method and PGPD method (the standard deviation of noise is $\sigma = 75$).

2.4.3 Comparison With the State-of-the-art Methods

We compare the proposed PGPD algorithm with BM3D [**bm3d**], EPLL [**epll**], LSSC [**lssc**], NCSR [**ncsr**], and WNNM [**wnnm**], which represent the state-of-the-arts of modern image denoising techniques and all of them exploit image NSS. The source codes of all competing algorithms are downloaded from the authors' websites and we use the default parameter settings.

To more clearly demonstrate the effectiveness of PG based NSS prior learning, we also compare with an extreme case of PGPD, i.e., letting $M = 1$ in the PG-GMM learning stage[1]. Clearly, this reduces to a patch based prior learning scheme and no NSS prior will be learned. We call this extreme case as *Patch Prior based Denoising* (PPD). The number of Gaussian components in PPD is set to 64, and the weighted sparse coding framework in it is the same as that in PGPD. All the other parameters in PPD are tuned to achieve its best performance.

2.4.4 Results and Discussions

We evaluate the competing methods from three aspects: PSNR, Speed, and Visual Quality.

PSNR. In Table 3.1, we present the PSNR results on four noise levels $\sigma = 30, 40, 50, 75$. The results on noise levels $\sigma = 10, 20, 100$ can be found in the supplementary material. From Table 3.1, we have several observations. Firstly, PGPD achieves much better PSNR results than PPD. The improvements are 0.24$\sim$0.52dB on average. This clearly demonstrates the effectiveness of PG-GMM in NSS prior learning. Secondly, PGPD has higher PSNR values than BM3D, LSSC, EPLL and NCSR, and is only slightly inferior to WNNM. However, PGPD is much more efficient than WNNM (see next paragraph). This validates the strong ability of PG based NSS prior in image denoising.

Speed. Efficiency is another important factor to evaluate an algorithm. We then compare the speed of all competing methods. All experiments are run under the Matlab2014b environment on a machine with Intel(R) Core(TM) i7-4770K CPU of 3.50GHz and 12.0 GB RAM. The run time (s) of competing methods on the test images is shown in Table 3.2. One can easily see that

[1] Since there is only 1 patch in the PG, the group mean vector cannot be subtracted and we subtract the mean value of the patch from it.

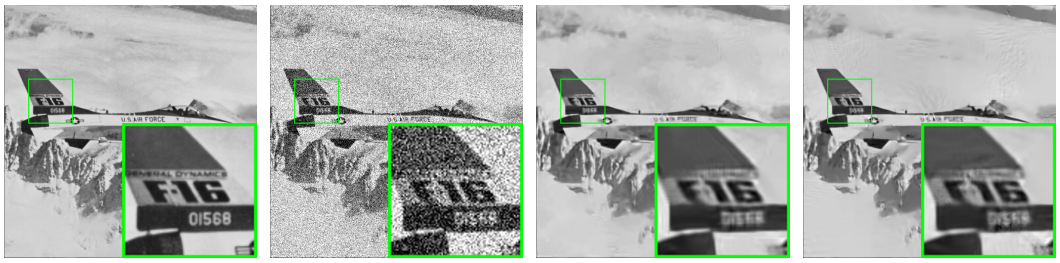
BM3D is the fastest method. The proposed PGPD is the second fastest, and it is much faster than the other methods. For a 256×256 image, BM3D costs about 0.8s while PGPD costs about 10s. However, please note that BM3D is implemented with compiled C++ mex-function and with parallelization, while PGPD is implemented purely in Matlab. EPLL is about 4 times slower than PGPD. Both LSSC and NCSR are very slow since they need to train online dictionary. Though WNNM has the highest PSNR, it suffers from huge computational cost due to the many online SVD operations. It is 10$\sim$16 times slower than PGPD.

Visual Quality. Considering that human subjects are the ultimate judge of the image quality, the visual quality of denoised images is also critical to evaluate a denoising algorithm. Fig. 3.6 and Fig. 3.7 show the denoised images of *Airplane* and *Cameraman* by the competing methods, respectively. Due to the page limit, the results of PPD are not shown here, and more visual comparisons can be found in the supplementary file. We can see that BM3D tends to over-smooth the image, while EPLL, LSSC, NCSR and WNNM are likely to generate artifacts when noise is high. Owe to the learned NSS prior, the proposed PGPD method is more robust against artifacts, and it preserves edge and texture areas much better than the other methods. For example, in image *Airplane*, PGPD reconstructs the numbers "01568" more clearly than all the other methods including WNNM. In image *Cameraman*, PGPD recoveries more faithfully the fine structures of the camera area.

In summary, the proposed PGPD method demonstrates powerful denoising ability quantitatively and qualitatively, and it is highly efficient.

2.5 Conclusion

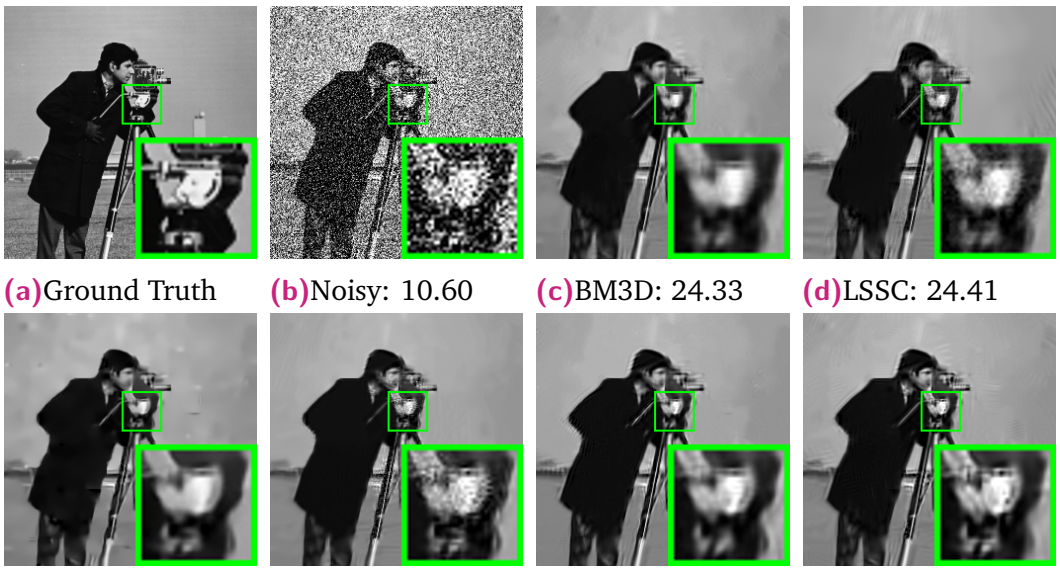
How to learn explicit models of nonlocal self-similarity (NSS) prior for image restoration is an open problem, and we made a good attempt on this by lifting the patch based image modeling to patch group (PG) based image modeling. A PG is a group of similar patches in an image region. After group mean subtraction, a PG can naturally represent the NSS variations of natural images. A PG based Gaussian Mixture Model (PG-GMM) learning algorithm was developed to learned the NSS prior from natural images, and an associated weighted sparse coding algorithm was developed for high performance image denoising. The so-called *PG Prior based Denoising*



(a)Ground Truth (b)Noisy: 14.16 (c)BM3D: 28.24 (d)LSSC: 28.15

(e)EPLL: 28.19 (f)NCSR: 28.18 (g)WNNM: 28.55 (h)PGPD: 28.38

Fig. 2.10: Denoised images and PSNR (dB) results of *Airplane* by different methods (the standard deviation of noise is $\sigma = 50$).



(a)Ground Truth (b)Noisy: 10.60 (c)BM3D: 24.33 (d)LSSC: 24.41

(e)EPLL: 24.29 (f)NCSR: 24.22 (g)WNNM: 24.55 (h)PGPD: 24.64

Fig. 2.11: Denoised images and PSNR (dB) results of *Cameraman* by different methods (the standard deviation of noise is $\sigma = 75$).

(PGPD) algorithm not only achieves highly competitive PSNR results with state-of-the-art denoising methods, but also is highly efficient and preserves better the image edges and textures. The proposed method can be extended to other image processing tasks such as deblurring and super-resolution.

Table 2.2: PSNR(dB) results of different denoising algorithms on 20 natural images.

	$\sigma = 30$							$\sigma = 40$						
Images	**BM3D**	**LSSC**	**EPLL**	**NCSR**	**WNNM**	**PPD**	**PGPD**	**BM3D**	**LSSC**	**EPLL**	**NCSR**	**WNNM**	**PPD**	**PGPD**
Airfield	26.41	26.68	26.52	26.36	26.67	26.33	26.46	25.10	25.51	25.36	25.07	25.48	25.20	25.30
Airplane	30.71	30.62	30.68	30.70	30.97	30.62	30.80	29.20	29.21	29.28	29.28	29.58	29.21	29.44
Baboon	24.57	24.78	24.70	24.63	24.85	24.54	24.63	23.11	23.51	23.35	23.28	23.58	23.23	23.39
Barbara	29.81	29.60	27.64	29.62	30.31	27.97	29.38	27.99	28.17	26.06	28.20	28.76	26.29	27.97
Boat	29.12	29.06	28.97	28.94	29.24	28.80	29.05	27.74	27.77	27.72	27.65	27.96	27.51	27.82
C. Man	28.64	28.63	28.40	28.58	28.80	28.25	28.53	27.18	27.34	27.10	27.12	27.47	27.05	27.33
Carhouse	28.78	28.79	28.70	28.72	28.94	28.62	28.80	27.38	27.49	27.38	27.40	27.58	27.29	27.51
Couple	28.87	28.76	28.69	28.57	28.98	28.54	28.84	27.48	27.41	27.34	27.24	27.62	27.16	27.53
Elaine	30.45	30.54	30.26	30.26	30.46	30.24	30.37	29.52	29.55	29.46	29.59	29.60	29.42	29.62
Hat	29.37	29.22	29.22	29.16	29.44	29.05	29.31	27.74	27.60	27.73	27.66	27.85	27.43	27.90
Hill	29.16	29.09	28.94	28.97	29.25	28.85	29.09	27.99	28.00	27.86	27.83	28.12	27.76	28.06
House	32.09	32.40	31.48	32.07	32.52	31.62	32.24	30.65	31.10	30.20	30.80	31.31	30.32	31.02
Lake	28.34	28.36	28.41	28.31	28.59	28.30	28.38	26.98	27.13	27.19	26.99	27.34	27.03	27.15
Leaves	27.81	27.65	27.36	28.14	28.60	27.51	27.99	25.69	26.04	25.80	26.24	26.95	25.88	26.29
Lena	31.26	31.18	30.98	31.06	31.43	30.98	31.27	29.86	29.91	29.69	29.92	30.11	29.67	30.10
Man	28.86	28.87	28.87	28.78	29.00	28.72	28.86	27.65	27.64	27.68	27.54	27.80	27.53	27.73
Monarch	28.36	28.20	28.50	28.46	28.91	28.27	28.49	26.72	26.87	27.05	26.85	27.47	26.81	27.02
Paint	28.29	28.29	28.45	28.10	28.58	28.39	28.42	26.69	26.77	27.00	26.50	27.10	26.88	26.94
Peppers	31.26	31.17	31.10	31.11	31.38	31.13	31.25	29.97	30.00	29.93	30.07	30.18	29.95	30.18
Zelda	30.45	30.27	30.44	30.16	30.48	30.35	30.43	29.10	28.91	29.18	28.94	29.12	29.07	29.23
Average	29.13	29.11	28.92	29.03	29.37	28.85	29.13	27.69	27.80	27.62	27.71	28.05	27.53	27.88

	$\sigma = 30$							$\sigma = 40$						
Images	**BM3D**	**LSSC**	**EPLL**	**NCSR**	**WNNM**	**PPD**	**PGPD**	**BM3D**	**LSSC**	**EPLL**	**NCSR**	**WNNM**	**PPD**	**PGPD**
Airfield	26.41	26.68	26.52	26.36	26.67	26.33	26.46	25.10	25.51	25.36	25.07	25.48	25.20	25.30
Airplane	30.71	30.62	30.68	30.70	30.97	30.62	30.80	29.20	29.21	29.28	29.28	29.58	29.21	29.44
Baboon	24.57	24.78	24.70	24.63	24.85	24.54	24.63	23.11	23.51	23.35	23.28	23.58	23.23	23.39
Barbara	29.81	29.60	27.64	29.62	30.31	27.97	29.38	27.99	28.17	26.06	28.20	28.76	26.29	27.97
Boat	29.12	29.06	28.97	28.94	29.24	28.80	29.05	27.74	27.77	27.72	27.65	27.96	27.51	27.82
C. Man	28.64	28.63	28.40	28.58	28.80	28.25	28.53	27.18	27.34	27.10	27.12	27.47	27.05	27.33
Carhouse	28.78	28.79	28.70	28.72	28.94	28.62	28.80	27.38	27.49	27.38	27.40	27.58	27.29	27.51
Couple	28.87	28.76	28.69	28.57	28.98	28.54	28.84	27.48	27.41	27.34	27.24	27.62	27.16	27.53
Elaine	30.45	30.54	30.26	30.26	30.46	30.24	30.37	29.52	29.55	29.46	29.59	29.60	29.42	29.62
Hat	29.37	29.22	29.22	29.16	29.44	29.05	29.31	27.74	27.60	27.73	27.66	27.85	27.43	27.90
Hill	29.16	29.09	28.94	28.97	29.25	28.85	29.09	27.99	28.00	27.86	27.83	28.12	27.76	28.06
House	32.09	32.40	31.48	32.07	32.52	31.62	32.24	30.65	31.10	30.20	30.80	31.31	30.32	31.02
Lake	28.34	28.36	28.41	28.31	28.59	28.30	28.38	26.98	27.13	27.19	26.99	27.34	27.03	27.15
Leaves	27.81	27.65	27.36	28.14	28.60	27.51	27.99	25.69	26.04	25.80	26.24	26.95	25.88	26.29
Lena	31.26	31.18	30.98	31.06	31.43	30.98	31.27	29.86	29.91	29.69	29.92	30.11	29.67	30.10
Man	28.86	28.87	28.87	28.78	29.00	28.72	28.86	27.65	27.64	27.68	27.54	27.80	27.53	27.73
Monarch	28.36	28.20	28.50	28.46	28.91	28.27	28.49	26.72	26.87	27.05	26.85	27.47	26.81	27.02
Paint	28.29	28.29	28.45	28.10	28.58	28.39	28.42	26.69	26.77	27.00	26.50	27.10	26.88	26.94
Peppers	31.26	31.17	31.10	31.11	31.38	31.13	31.25	29.97	30.00	29.93	30.07	30.18	29.95	30.18
Zelda	30.45	30.27	30.44	30.16	30.48	30.35	30.43	29.10	28.91	29.18	28.94	29.12	29.07	29.23
Average	29.13	29.11	28.92	29.03	29.37	28.85	29.13	27.69	27.80	27.62	27.71	28.05	27.53	27.88

	$\sigma = 50$							$\sigma = 75$						
Images	**BM3D**	**LSSC**	**EPLL**	**NCSR**	**WNNM**	**PPD**	**PGPD**	**BM3D**	**LSSC**	**EPLL**	**NCSR**	**WNNM**	**PPD**	**PGPD**
Airfield	31.32	31.51	31.37	31.40	31.47	31.18	28.13	28.48	28.18	28.07	28.40	28.19		
Airplane	28.24	28.15	28.19	28.18	28.55	28.10	28.38	26.40	26.16	26.14	26.10	26.68	25.90	26.39
Baboon	22.35	22.60	22.35	22.43	22.73	22.30	22.47	21.11	21.18	20.85	21.03	21.36	20.71	21.09
Barbara	27.23	27.03	24.83	26.99	27.79	24.94	26.81	25.12	25.01	22.94	24.72	25.81	22.84	24.84
Boat	26.78	26.77	26.74	26.67	26.97	26.52	26.85	25.12	25.03	25.01	24.87	25.29	24.72	25.19
C. Man	26.12	26.35	26.10	26.15	26.42	26.13	26.46	24.33	24.41	24.29	24.22	24.55	24.36	24.64
Carhouse	26.53	26.48	26.39	26.41	26.67	26.27	26.53	24.89	24.85	24.65	24.53	25.04	24.44	24.85
Couple	26.46	26.35	26.30	26.19	26.65	26.07	26.50	24.70	24.51	24.51	24.33	24.85	24.22	24.70
Elaine	28.94	28.75	28.77	28.85	28.97	28.69	28.90	27.41	27.27	27.38	27.16	27.53	27.26	27.47
Hat	26.77	26.41	26.62	26.51	26.78	26.28	26.76	24.77	24.31	24.65	24.48	24.77	24.19	24.79
Hill	27.19	27.14	27.04	26.99	27.34	26.91	27.22	25.68	25.57	25.60	25.40	25.88	25.34	25.73
House	29.69	29.99	29.12	29.62	30.32	29.17	29.93	27.51	27.75	27.09	27.22	28.25	26.81	27.81
Lake	26.13	26.15	26.24	26.02	26.41	26.05	26.20	24.49	24.25	24.50	24.26	24.66	24.19	24.49
Leaves	24.68	24.78	24.55	24.96	25.47	24.56	25.03	22.49	22.17	22.12	22.60	23.06	21.94	22.61
Lena	29.05	28.95	28.68	28.90	29.25	28.61	29.11	27.26	27.22	26.88	27.00	27.54	26.68	27.40
Man	26.81	26.72	26.79	26.67	26.94	26.63	26.86	25.32	25.10	25.26	25.10	25.42	25.01	25.36
Monarch	25.82	25.88	25.94	25.76	26.31	25.66	26.00	23.91	23.66	23.88	23.67	24.31	23.51	24.00
Paint	25.67	25.59	25.87	25.36	25.98	25.70	25.82	23.80	23.52	23.88	23.44	24.07	23.50	23.89
Peppers	29.12	29.06	28.98	29.07	29.34	28.99	29.22	27.28	27.14	27.15	26.96	27.55	27.04	27.42
Zelda	28.25	27.90	28.22	27.97	28.21	28.06	28.24	26.60	26.09	26.55	26.21	26.44	26.37	26.56
Average	26.80	26.78	26.61	26.69	27.08	26.50	26.89	25.04	24.90	24.81	24.79	25.30	24.59	25.11

Table 2.3: Average run time (seconds) with standard deviation of different methods on images of size 256×256 and 512×512. BM3D uses parallelization and is implemented with compiled C++ mex-function while the other methods are implemented in Matlab.

	256×256						
σ	**BM3D**	**LSSC**	**EPLL**	**NCSR**	**WNNM**	**PPD**	**PGPD**
10	0.67 ± 0.09	186.90 ± 4.02	38.47 ± 0.10	126.43 ± 3.84	84.34 ± 1.42	10.15 ± 0.07	8.00 ± 0.05
20	0.70 ± 0.09	184.21 ± 5.82	38.47 ± 0.13	156.14 ± 5.26	84.70 ± 1.71	10.18 ± 0.15	8.09 ± 0.09
30	0.70 ± 0.09	212.07 ± 8.72	38.55 ± 0.09	149.31 ± 4.19	155.75 ± 0.94	10.34 ± 0.25	8.47 ± 0.07
40	0.67 ± 0.11	209.13 ± 6.99	38.51 ± 0.08	346.91 ± 18.65	157.35 ± 1.48	10.47 ± 0.21	9.80 ± 0.08
50	0.87 ± 0.04	221.36 ± 6.27	40.21 ± 1.82	326.93 ± 9.64	119.47 ± 4.65	10.88 ± 0.05	9.91 ± 0.13
75	0.89 ± 0.03	240.75 ± 6.08	40.91 ± 1.33	258.04 ± 11.80	179.30 ± 5.08	10.87 ± 0.27	11.73 ± 0.08
100	0.90 ± 0.03	257.25 ± 6.01	42.80 ± 1.93	252.74 ± 8.50	191.32 ± 1.47	10.90 ± 0.19	11.78 ± 0.08
	512×512						
σ	**BM3D**	**LSSC**	**EPLL**	**NCSR**	**WNNM**	**PPD**	**PGPD**
10	3.16 ± 0.12	746.53 ± 24.96	160.93 ± 2.81	624.83 ± 40.24	352.34 ± 3.87	41.79 ± 0.32	33.03 ± 0.25
20	3.32 ± 0.11	762.62 ± 31.25	159.80 ± 0.37	751.09 ± 42.89	351.09 ± 3.14	42.09 ± 0.41	33.26 ± 0.29
30	3.32 ± 0.09	856.82 ± 40.32	160.21 ± 0.18	709.90 ± 31.62	650.54 ± 7.23	42.36 ± 0.99	35.45 ± 0.24
40	3.18 ± 0.18	865.83 ± 40.96	160.23 ± 0.17	1620.74 ± 104.59	652.49 ± 10.49	41.70 ± 0.47	40.13 ± 0.23
50	3.85 ± 0.09	891.53 ± 48.60	161.36 ± 3.08	1492.78 ± 65.87	476.50 ± 12.34	41.75 ± 0.64	40.40 ± 0.28
75	3.91 ± 0.05	983.05 ± 69.96	165.66 ± 2.62	1156.82 ± 66.37	784.92 ± 18.32	41.88 ± 0.78	50.00 ± 0.25
100	3.94 ± 0.04	1087.57 ± 68.76	177.51 ± 7.16	1100.00 ± 26.64	824.56 ± 34.41	42.80 ± 1.09	50.32 ± 0.31

3 External Prior Guided Internal Prior Learning for Real Noisy Image Denoising

> *Innovation distinguishes between a leader and a follower.*
>
> **— Steve Jobs**
> (CEO Apple Inc.)

3.1 Learning External Nonlocal Self-Similarity Priors

Image denoising is a crucial and indispensable step to improve image quality in digital imaging systems. In particular, with the decrease of size of CMOS/CCD sensors, image is more easily to be corrupted by noise and hence denoising is becoming increasingly important for high resolution imaging. The problem of image denoising has been extensively studied in literature and numerous image denoising methods [**bayesshrink**; **curvelet**; **ksvd**; **lssc**; **ncsr**; **bm3d**; **cbm3d**; **zhou2012nonparametric**; **Tomasi1998**; **blsgsm**; **nlm**; **nlbayes**; **wnnm**; **pgpd**; **foe**; **epll**; **mlp**; **xie2012image**; **zhang2017beyond**; **barbu2009training**; **csf**; **chen2015learning**; **Fadili**; **salmon2014**; **Foipractical**; **Luisier**; **Makitalo2013Optimal**; **Montagner**; **jiang2014mixed**; **Hu2016**; **xuaccv2016**; **fullyblind**; **rabie2005robust**; **Liu2008**; **almapg**; **noiseclinic**; **ncwebsite**; **Zhu_2016_CVPR**; **crosschannel2016**; **neatimage**] have been proposed in the past decades. Most of existing denoising methods focus on the scenario of additive white Gaussian noise (AWGN) [**bayesshrink**; **curvelet**; **ksvd**; **lssc**; **ncsr**; **bm3d**; **cbm3d**; **zhou2012nonparametric**; **Tomasi1998**; **blsgsm**; **nlm**; **nlbayes**; **wnnm**; **pgpd**; **foe**; **epll**; **mlp**; **xie2012image**; **zhang2017beyond**; **barbu2009training**; **csf**; **chen2015learning**], where the observed noisy image $\mathbf{y}$ is modeled as the addition of clean image $\mathbf{x}$ and AWGN $\mathbf{n}$, i.e., $\mathbf{y} = \mathbf{x} + \mathbf{n}$. There are also methods proposed for removing Poisson noise [**Fadili**; **salmon2014**], mixed Poisson and Gaussian noise [**Foipractical**; **Luisier**;

Makitalo2013Optimal; **Montagner**], mixed Gaussian and impulse noise [**jiang2014mixed**; **Hu2016**; **xuaccv2016**], and realistic noise in real photography [**fullyblind**; **rabie2005robust**; **Liu2008**; **almapg**; **Zhu_2016_CVPR**; **noiseclinic**; **ncwebsite**; **crosschannel2016**; **neatimage**].

Natural images have many properties, such as sparsity and nonlocal self-similarity, which can be employed as useful priors for designing image denoising methods. Based on the facts that natural images will be sparsely distributed in some transformed domain, wavelet [**bayesshrink**] and curvelet [**curvelet**] transforms have been widely adopted for image denoising. The sparse representation based methods [**ksvd**; **lssc**; **ncsr**; **bm3d**; **cbm3d**; **zhou2012nonparametric**] encode image patches over a dictionary by using ℓ_1-norm minimization to enforce the sparsity. The well-known bilateral filters [**Tomasi1998**] employ the prior information that image pixels exhibit similarity in both spatial domain and intensity domain. Other image priors such as multiscale self-similarity [**blsgsm**] and nonlocal self-similarity [**nlm**; **nlbayes**], or the combination of multiple image priors [**wnnm**; **pgpd**] have also been successfully used in image denoising. For example, by using low-rank minimization to characterize the image nonlocal self-similarity, the WNNM [**wnnm**] method achieves state-of-the-art performance for AWGN denoising.

Instead of using predefined image priors, methods have also been proposed to learn priors from natural images for denoising. The generative image prior learning methods usually learn prior models from a set of external clean images and apply the learned prior models to the given noisy image [**foe**; **epll**; **pgpd**], or learn priors from the given noisy image to perform denoising [**ksvd**]. Recently, the discriminative image prior learning methods [**mlp**; **xie2012image**; **zhang2017beyond**; **chen2015learning**; **barbu2009training**; **csf**], which learn denoising models from pairs of clean and noisy images, have been becoming popular. The representative methods include the neural network based methods [**mlp**; **xie2012image**; **zhang2017beyond**], random fields based methods [**barbu2009training**; **csf**], and reaction diffusion based methods [**chen2015learning**].

Most of the above mentioned methods focus on AWGN removal, however, the assumption of AWGN is too ideal to be true for real-world noisy images, where the noise is much more complex and varies with different scenes, cameras and camera settings (ISO, shutter speed, and aperture, etc.) [**crosschannel2016**;

(a)Noisy 33.30 (b)CBM3D34.55 (c)WNNM35.85 (d)CSF 35.39 (e)TNRD 35.97

(f)DnCNN34.14 (g)NI 34.39 (h)NC 35.33 (i)Ours **37.49** (j)Mean Image

Fig. 3.1: Denoised images and PSNR (dB) results of a region cropped from the real noisy image "Nikon D800 ISO 3200 A3" [**crosschannel2016**] by different methods. The scene was shot 500 times with the same camera and camera setting. The mean image of the 500 shots is roughly taken as the "ground truth", with which the PSNR index can be computed. The images are better viewed by zooming in on screen.

healey1994radiometric]. As a result, many denoising methods in literature, including those learning based methods, become less effective when applied to real-world noisy images. Fig. 4.1 shows an example, where we apply some representative and state-of-the-art denoising methods, including CBM3D [**cbm3d**], WNNM [**wnnm**], DnCNN [**zhang2017beyond**], CSF [**csf**], and TNRD [**chen2015learning**] to a real noisy image (captured by a Nikon D800 camera with ISO is 3200) provided in [**crosschannel2016**]. One can see that these methods either remain much the noise or over-smooth the image details.

There have been a few methods [**fullyblind**; **rabie2005robust**; **Liu2008**; **almapg**; **crosschannel2016**; **Zhu_2016_CVPR**; **noiseclinic**; **ncwebsite**] and software toolboxes [**neatimage**] developed for real noisy image denoising. Almost all of these methods follow a two-stage framework: first estimate the parameters of the noise model (usually assumed to be Gaussian or mixture of Gaussians (MoG)), and then perform denoising with the estimated noise model. However, the noise in real noisy images is very complex and is hard to be modeled by explicit distributions such as Gaussian and MoG. According to [**healey1994radiometric**], the noise corrupted in the in-camera imaging process [**tsin2001statistical**; **NewInCamera**; **crosschannel2016**; **karaimer_brown_ECCV_2016**] is signal dependent and comes from five main sources: photon shot, fixed pattern, dark current, readout, and quanti-

zation noise. The existing methods [**fullyblind**; **rabie2005robust**; **Liu2008**; **almapg**; **crosschannel2016**; **Zhu_2016_CVPR**; **noiseclinic**; **ncwebsite**; **neatimage**] mentioned above may not perform well on real noisy image denoising tasks. Fig. 4.1 also shows the denoising results of two real noisy image denoising methods, Noise Clinic [**noiseclinic**; **ncwebsite**] and Neat Image [**neatimage**]. One can see that these two methods still generate much noise caused artifacts.

This work aims to develop a new paradigm for real noisy image denoising. Different from existing real noisy image denoising methods [**fullyblind**; **rabie2005robust**; **Liu2008**; **almapg**; **crosschannel2016**; **Zhu_2016_CVPR**; **noiseclinic**; **ncwebsite**] which focus on noise modeling, we focus on image prior learning. We argue that with a strong and adaptive prior learning scheme, robust denoising performance on real noisy images can still be obtained. To achieve this goal, we propose to first learn image priors from external clean images, and then employ the learned external priors to guide the learning of internal priors from the given noisy image. The flowchart of the proposed method is illustrated in Fig. 2. We first extract millions of patch groups (PGs) from a set of high quality natural images, with which a Gaussian Mixture Model (GMM) is learned as the external image prior. The learned GMM prior model is used to cluster the PGs extracted from the given noisy image, and then an external-internal hybrid orthogonal dictionary is learned as the final prior for each cluster, with which the denoising can be readily performed by weighted sparse coding with closed form solution. Our proposed denoising method is simple and efficient, yet our extensive experiments on real noisy images demonstrate its better denoising performance than the current state-of-the-arts.

3.2 Related Work

3.2.1 Internal and External Prior Learning

Learning natural image priors plays a key role in image denoising [**blsgsm**; **zhou2012nonparametric**; **ksvd**; **lssc**; **ncsr**; **foe**; **epll**; **pgpd**; **mlp**; **xie2012image**; **zhang2017beyond**; **barbu2009training**; **csf**; **chen2015learning**]. There are mainly three categories of prior learning based methods. 1) External prior learning methods [**foe**; **epll**; **pgpd**] learn priors (e.g., dictionaries) from a set of external clean images, and the learned priors are used to recover the latent

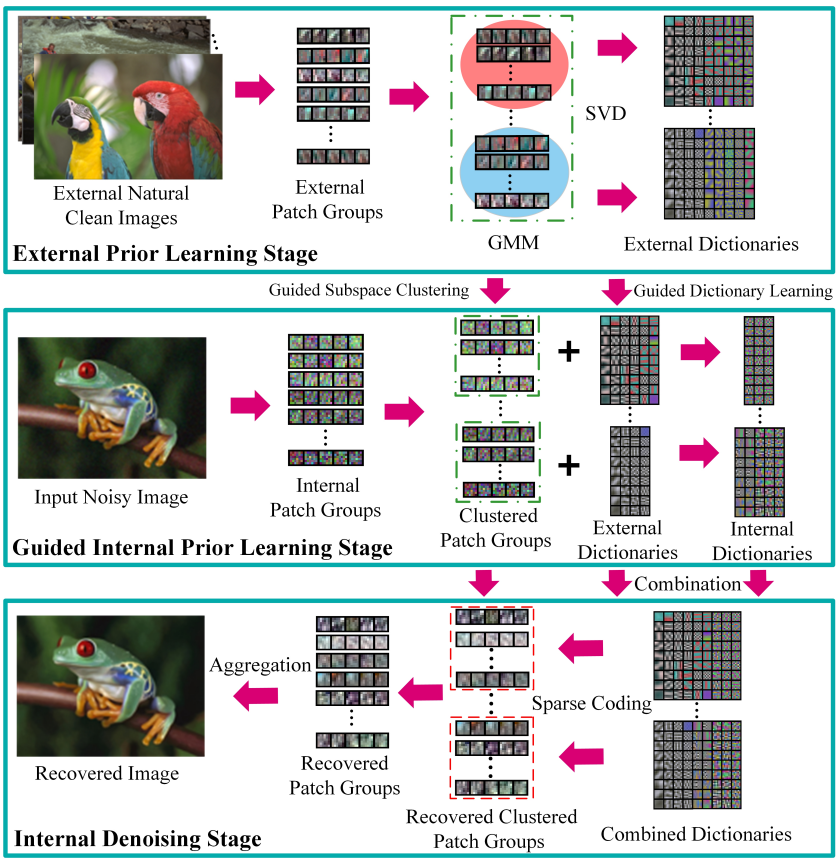


Fig. 3.2: Flowchart of the proposed external prior guided internal prior learning and denoising framework.

clean image from the given noisy image. 2) Internal prior learning methods [**blsgsm**; **zhou2012nonparametric**; **ksvd**; **lssc**; **ncsr**] directly learn priors from a given noisy image, and image denoising is often done simultaneously with the prior learning process. 3) Discriminative prior learning methods [**mlp**; **xie2012image**; **zhang2017beyond**; **barbu2009training**; **csf**; **chen2015learning**] learn discriminative models or mapping functions from clean and noisy image pairs, and the learned models or mapping functions are applied to a noisy image for denoising. This paper focuses on the first two categories of prior learning methods, which belongs to generative prior learning methods.

It has been shown [**foe**; **epll**; **pgpd**] that the external priors learned from natural clean images are effective and efficient for universal image denoising problems, whereas they are not adaptive to the given noisy image and some fine-scale image structures may not be well recovered. By contrast, the internal priors learned from the given noisy image are adaptive to image content, but the learned priors can be much affected by noise and the learning processing is usually slow. In addition, most of the internal prior learning methods [**blsgsm**; **zhou2012nonparametric**; **ksvd**; **lssc**; **ncsr**] assume AWGN noise, making the learned priors less robust for real noisy images. In this paper, we use external priors to guide the internal prior learning. Our method is not only much faster than the traditional internal learning methods, but also very robust to denoise real noisy images.

3.2.2 Real Noisy Image Denoising

Most of the denoising methods in literature [**bayesshrink**; **curvelet**; **ksvd**; **lssc**; **ncsr**; **bm3d**; **cbm3d**; **zhou2012nonparametric**; **Tomasi1998**; **blsgsm**; **nlm**; **nlbayes**; **wnnm**; **pgpd**; **foe**; **epll**; **mlp**; **xie2012image**; **zhang2017beyond**; **barbu2009training**; **csf**; **chen2015learning**] assume AWGN noise and use simulated noisy images for algorithm design and evaluation. Recently, several denoising methods have been proposed to remove unknown noise from real-world noisy images [**fullyblind**; **rabie2005robust**; **Liu2008**; **almapg**; **crosschannel2016**; **Zhu_2016_CVPR**; **noiseclinic**; **ncwebsite**]. Portilla [**fullyblind**] employed a correlated Gaussian model to estimate the noise of each wavelet subband. Rabie [**rabie2005robust**] modeled the noisy pixels as outliers and performed denoising via Lorentzian robust estimator. Liu et al. [**Liu2008**] proposed the "noise level function" to estimate the noise and performed denoising by learning a Gaussian conditional random field. Gong et al.

[**almapg**] proposed to model the data fitting term via weighted sum of ℓ_1 and ℓ_2 norms and performed denoising by a simple sparsity regularization term in the wavelet transform domain. The "Noise Clinic" [**noiseclinic**; **ncwebsite**] estimates the noise distribution by using a multivariate Gaussian model and removes the noise by using a generalized version of nonlocal Bayesian model [**nlbayes**]. Zhu et al. [**Zhu_2016_CVPR**] proposed a Bayesian method to approximate and remove the noise via a low-rank mixture of Gaussians (MoG) model. The method in [**crosschannel2016**] models the cross-channel noise in real noisy image as a multivariate Gaussian and the noise is removed by the Bayesian nonlocal means filter [**kervrann2007bayesian**]. The commercial software Neat Image [**neatimage**] estimates the noise parameters from a flat region of the given noisy image and filters the noise correspondingly.

The methods [**fullyblind**; **rabie2005robust**; **Liu2008**; **almapg**; **crosschannel2016**; **Zhu_2016_CVPR**; **noiseclinic**; **ncwebsite**] mentioned above emphasize much on the noise modeling, and they use Gaussian or MoG to model the noise in real noisy images. Nonetheless, the noise in real noisy images is very complex and hard to be modeled by explicit distributions [**healey1994radiometric**]. These works ignore the importance of learning image priors, which actually can be easier to model compared with modeling the complex realistic noise. In this paper, we propose a simple yet effective image prior learning method for real noisy image denoising. Due to its strong prior modeling ability, the proposed method simply models the noise as locally Gaussian, and it achieves highly competitive performance on real noisy image denoising.

3.3 External Prior Guided Internal Prior Learning for Image Denoising

In this section, we first describe the learning of external prior, and then describe in detail the guided internal prior learning method, followed by the denoising algorithm.

3.3.1 Learn External Patch Group Priors

The nonlocal self-similarity based patch group (PG) prior learning [**pgpd**] has proved to be very effective for image denosing. In this work, we extract

PGs from natural clean images to learn external priors. A PG is a group of similar patches to a local patch. In our method, each local patch is extracted from a RGB image with patch size $p \times p \times 3$. We search the M most similar (i.e., smallest Euclidean distance) patches to this local patch (including the local patch itself) in a $W \times W$ region around it. Each patch is stretched to a patch vector $\mathbf{x}_m \in \mathbb{R}^{3p^2 \times 1}$ to form the PG, denoted by $\{\mathbf{x}_m\}_{m=1}^M$. The mean vector of this PG is $\boldsymbol{\mu} = \frac{1}{M}\sum_{m=1}^M \mathbf{x}_m$, and the group mean subtracted PG is defined as $\overline{\mathbf{X}} \triangleq \{\overline{\mathbf{x}}_m = \mathbf{x}_m - \boldsymbol{\mu}\}_{m=1}^M$.

Assume that a number of L PGs are extracted from a set of external natural images, and the l-th PG is $\overline{\mathbf{X}}_l \triangleq \{\overline{\mathbf{x}}_{l,m}\}_{m=1}^M, l = 1, ..., L$. A Gaussian Mixture Model (GMM) is learned to model the PG prior. The overall log-likelihood function is

$$\ln \mathcal{L} = \sum_{l=1}^{L} \ln(\sum_{k=1}^{K} \pi_k \prod_{m=1}^{M} \mathcal{N}(\overline{\mathbf{x}}_{l,m}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)). \tag{3.1}$$

The learning process is similar to the GMM learning in [**pgpd**; **epll**]. Finally, a GMM model with K Gaussian components is learned, and the learned parameters include mixture weights $\{\pi_k\}_{k=1}^K$, mean vectors $\{\boldsymbol{\mu}_k\}_{k=1}^K$, and covariance matrices $\{\boldsymbol{\Sigma}_k\}_{k=1}^K$. Note that the mean vector of each cluster is naturally zero, i.e., $\boldsymbol{\mu}_k = \mathbf{0}$.

To better describe the subspace of each Gaussian component, we perform singular value decomposition (SVD) [**eckart1936approximation**] on the covariance matrix:

$$\boldsymbol{\Sigma}_k = \boldsymbol{U}_k \boldsymbol{S}_k \boldsymbol{U}_k^\top. \tag{3.2}$$

The eigenvector matrices $\{\boldsymbol{U}_k\}_{k=1}^K$ will be employed as the external orthogonal dictionary to guide the internal sub-dictionary learning in next sub-section. The singular values in $\boldsymbol{S}_k$ reflect the significance of the singular vectors in $\boldsymbol{U}_k$. They will also be utilized as prior weights for weighted sparse coding in our denoising algorithm.

3.3.2 Guided Internal Prior Learning

After the external PG prior model is learned from external natural clean images, we employ it to guide the internal PG prior learning for a given real noisy image. The guidance lies in two aspects. First, the external prior will guide the subspace clustering of internal noisy PGs. Second, the external prior will guide the orthogonal dictionary learning of internal noisy PGs.

Internal Subspace Clustering

Given a real noisy image $\mathbf{y}$, we extract N (overlapped) local patches from it. Similar to the external prior learning stage, for the n-th ($n = 1, ..., N$) local patch we search its M most similar (by Euclidean distance) patches around it to form a noisy PG, denoted by $\boldsymbol{Y}_n = \{\mathbf{y}_{n,1}, ..., \mathbf{y}_{n,M}\}$. Then the group mean of $\boldsymbol{Y}_n$, denoted by $\boldsymbol{\mu}_n$, is subtracted from each patch by $\overline{\boldsymbol{y}}_{n,m} \triangleq \mathbf{y}_{n,m} - \boldsymbol{\mu}_n$, leading to the mean subtracted noisy PG $\overline{\boldsymbol{Y}}_n \triangleq \{\overline{\boldsymbol{y}}_{n,m}\}_{m=1}^{M}$.

The external GMM prior models $\{\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_k)\}_{k=1}^{K}$ basically characterize the subspaces of natural high quality PGs. Therefore, we project each noisy PG $\overline{\boldsymbol{Y}}_n$ into the subspaces of $\{\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_k)\}_{k=1}^{K}$ and assign it to the most suitable subspace based on the posterior probability:

$$P(k|\overline{\boldsymbol{Y}}_n) = \frac{\prod_{m=1}^{M} \mathcal{N}(\overline{\boldsymbol{y}}_{n,m}|\mathbf{0}, \boldsymbol{\Sigma}_k)}{\sum_{l=1}^{K} \prod_{m=1}^{M} \mathcal{N}(\overline{\boldsymbol{y}}_{n,m}|\mathbf{0}, \boldsymbol{\Sigma}_l)} \tag{3.3}$$

for $k = 1, ..., K$. Then $\overline{\boldsymbol{Y}}_n$ is assigned to the subspace with the maximum a-posteriori (MAP) probability $\max_k P(k|\overline{\boldsymbol{Y}}_n)$.

Guided Orthogonal Dictionary Learning

Assume that we have assigned all the internal noisy PGs $\{\overline{\boldsymbol{Y}}_n\}_{n=1}^{N}$ to their corresponding most suitable subspaces in $\{\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_k)\}_{k=1}^{K}$. For the k-th subspace, the noisy PGs assigned to it are $\{\overline{\boldsymbol{Y}}_{k_n}\}_{n=1}^{N_k}$, where $\overline{\boldsymbol{Y}}_{k_n} = [\overline{\boldsymbol{y}}_{k_n,1}, ..., \overline{\boldsymbol{y}}_{k_n,M}]$ and $\sum_{k=1}^{K} N_k = N$. We propose to learn an orthogonal dictionary $\boldsymbol{D}_k$ from each set of PGs $\overline{\boldsymbol{Y}}_{k_n}$ to characterize the internal PG prior with the guidance of the corresponding external orthogonal dictionary $\boldsymbol{U}_k$ (Eq. (3.2)). The reasons that we learn orthogonal dictionaries are two-fold. Firstly, the PGs $\{\overline{\boldsymbol{Y}}_{k_n}\}_{n=1}^{N_k}$ are in a subspace of the whole space of all PGs; therefore, there is no necessary to learn a redundant over-complete dictionary to characterize it, while an orthonormal dictionary has naturally zero *mutual incoherence* [**donoho2001uncertainty**]. Secondly, the orthogonality of dictionary can make the patch encoding in the testing stage very efficient, leading to an efficient denoising algorithm (please refer to sub-section III-C for more details).

We let the orthogonal dictionary $\boldsymbol{D}_k$ be

$$\boldsymbol{D}_k \triangleq [\boldsymbol{D}_{k,\mathrm{E}}\ \boldsymbol{D}_{k,\mathrm{I}}] \in \mathbb{R}^{3p^2 \times 3p^2}, \tag{3.4}$$

where $\boldsymbol{D}_{k,\mathrm{E}} = \boldsymbol{U}_k(:, 1:r) \in \mathbb{R}^{3p^2 \times r}$ is the external sub-dictionary and it includes the first r most important eigenvectors of $\boldsymbol{U}_k$, and the internal sub-dictionary $\boldsymbol{D}_{k,\mathrm{I}} \in \mathbb{R}^{3p^2 \times (3p^2 - r)}$ is to be adaptively learned from the noisy PGs $\{\overline{\boldsymbol{Y}}_{k_n}\}_{n=1}^{N_k}$. The rationale to design $\boldsymbol{D}_k$ as a hybrid dictionary is as follows. The external sub-dictionary $\boldsymbol{D}_{k,\mathrm{E}}$ is pre-trained from external clean data, and it represents the k-th latent subspace of natural images, which is helpful to reconstruct the common latent structures of images. However, $\boldsymbol{D}_{k,\mathrm{E}}$ is general to all images but not adaptive to the given noisy image. Some fine-scale details specific to the given image may not be well characterized by $\boldsymbol{D}_{k,\mathrm{E}}$. Therefore, we learn an internal sub-dictionary $\boldsymbol{D}_{k,\mathrm{I}}$ to supplement $\boldsymbol{D}_{k,\mathrm{E}}$. In other words, $\boldsymbol{D}_{k,\mathrm{I}}$ is to reveal the latent subspace adaptive to the input noisy image, which cannot be effectively represented by $\boldsymbol{D}_{k,\mathrm{E}}$.

For notation simplicity, in the following development we ignore the subspace index k for $\overline{\boldsymbol{Y}}_{k_n}$ and $\boldsymbol{D}_k$, etc. The learning of hybrid orthogonal dictionary $\boldsymbol{D}$ is performed under the following weighted sparse coding framework:

$$\begin{aligned} \min_{\boldsymbol{D}_{\mathrm{I}}, \{\boldsymbol{\alpha}_{n,m}\}} & \sum_{n=1}^{N} \sum_{m=1}^{M} (\|\overline{\boldsymbol{y}}_{n,m} - \boldsymbol{D}\boldsymbol{\alpha}_{n,m}\|_2^2 + \sum_{j=1}^{3p^2} \lambda_j |\boldsymbol{\alpha}_{n,m,j}|) \\ \text{s.t.} \quad & \boldsymbol{D} = [\boldsymbol{D}_{\mathrm{E}}\ \boldsymbol{D}_{\mathrm{I}}],\ \boldsymbol{D}^\top \boldsymbol{D} = \boldsymbol{I}, \end{aligned} \tag{3.5}$$

where $\boldsymbol{I}$ is the $3p^2$ dimensional identity matrix, $\boldsymbol{\alpha}_{n,m}$ is the sparse coding vector of the m-th patch $\overline{\boldsymbol{y}}_{n,m}$ in the n-th PG $\overline{\boldsymbol{Y}}_n$ and $\boldsymbol{\alpha}_{n,m,j}$ is the j-th element of $\boldsymbol{\alpha}_{n,m}$. λ_j is the j-th regularization parameter defined as

$$\lambda_j = \lambda / (\sqrt{\boldsymbol{S}_k(j)} + \varepsilon), \tag{3.6}$$

where $\boldsymbol{S}_k(j)$ is the j-th singular value of diagonal singular value matrix $\boldsymbol{S}_k$ (please refer to Eq. (3.2)) and ε is a small positive number to avoid zero denominator. Note that $\boldsymbol{D}_{\mathrm{E}} = \boldsymbol{U}_k$ if $r = 3p^2$ and $\boldsymbol{D}_{\mathrm{E}} = \emptyset$ if $r = 0$.

In the dictionary learning model (3.5), we use the ℓ_2 norm to model the representation residual of PGs. This is because the patches in those PGs have similar content, and we assume that the noise therein will have similar statistics, which can be roughly modeled as locally Gaussian. On the other hand, this will make the dictionary learning much easier to solve. We employ an alternating iterative approach to solve the optimization problem (3.5). Specifically, we initialize the orthogonal dictionary as $\boldsymbol{D}^{(0)} = \boldsymbol{U}_k$ and for $t = 0, 1, ..., T-1$, and alternatively update $\boldsymbol{\alpha}_{n,m}$ and $\boldsymbol{D}_{\mathrm{I}}$ as follows.

Updating Sparse Coding Coefficients: Given the orthogonal dictionary $\boldsymbol{D}^{(t)}$, we update each sparse coding vector $\boldsymbol{\alpha}_{n,m}$ by solving

$$\boldsymbol{\alpha}_{n,m}^{(t+1)} := \arg\min_{\boldsymbol{\alpha}_{n,m}} \|\overline{\boldsymbol{y}}_{n,m} - \boldsymbol{D}^{(t)}\boldsymbol{\alpha}_{n,m}\|_2^2 + \sum_{j=1}^{3p^2} \lambda_j |\boldsymbol{\alpha}_{n,m,j}|. \tag{3.7}$$

Since dictionary $\boldsymbol{D}^{(t)}$ is orthogonal, the problems (3.7) has a closed-form solution

$$\boldsymbol{\alpha}_{n,m}^{(t+1)} = \text{sgn}((\boldsymbol{D}^{(t)})^\top \overline{\boldsymbol{y}}_{n,m}) \odot \max(|(\boldsymbol{D}^{(t)})^\top \overline{\boldsymbol{y}}_{n,m}| - \boldsymbol{\lambda}, \boldsymbol{0}), \tag{3.8}$$

where $\boldsymbol{\lambda} = \frac{1}{2}[\lambda_1, \lambda_2, ..., \lambda_{3p^2}]^\top$ is the vector of regularization parameter, sgn$(\bullet)$ is the sign function and $\odot$ means element-wise multiplication. The detailed derivation of Eq. (3.8) can be found in Appendix A.

Updating Internal Sub-dictionary: Given the sparse coding vectors $\{\boldsymbol{\alpha}_{n,m}^{(t+1)}\}$, we update the internal sub-dictionary by solving

$$\begin{aligned} \boldsymbol{D}_{\text{I}}^{(t+1)} :&= \arg\min_{\mathbf{D}_{\text{I}}} \sum_{n=1}^{N}\sum_{m=1}^{M} \|\overline{\boldsymbol{y}}_{n,m} - \boldsymbol{D}\boldsymbol{\alpha}_{n,m}^{(t+1)}\|_2^2 \\ &= \arg\min_{\mathbf{D}_{\text{I}}} \|\overline{\boldsymbol{Y}}_n - \boldsymbol{D}\boldsymbol{A}^{(t+1)}\|_F^2 \\ \text{s.t.} \quad \boldsymbol{D} &= [\boldsymbol{D}_{\text{E}}\ \boldsymbol{D}_{\text{I}}],\ \boldsymbol{D}_{\text{I}}^\top \boldsymbol{D}_{\text{I}} = \boldsymbol{I}_{(3p^2-r)},\ \boldsymbol{D}_{\text{E}}^\top \boldsymbol{D}_{\text{I}} = \boldsymbol{0}, \end{aligned} \tag{3.9}$$

where $\mathbf{A}^{(t+1)} = [\boldsymbol{\alpha}_{1,1}^{(t+1)}, ..., \boldsymbol{\alpha}_{1,M}^{(t)}, ..., \boldsymbol{\alpha}_{N,1}^{(t+1)}, ..., \boldsymbol{\alpha}_{N,M}^{(t+1)}]$ and $\boldsymbol{I}_{(3p^2-r)}$ is the $(3p^2-r)$ dimensional identity matrix. The sparse coefficients matrix can be written as $\boldsymbol{A}^{(t+1)} = [(\boldsymbol{A}_{\text{E}}^{(t+1)})^\top\ (\boldsymbol{A}_{\text{I}}^{(t+1)})^\top]^\top$ where the external part $\boldsymbol{A}_{\text{E}}^{(t+1)} \in \mathbb{R}^{r\times NM}$ and the internal part $\boldsymbol{A}_{\text{I}}^{(t+1)} \in \mathbb{R}^{(3p^2-r)\times NM}$ represent the coding coefficients of $\boldsymbol{Y}$ over external sub-dictionary $\boldsymbol{D}_{\text{E}}$ and internal sub-dictionary $\boldsymbol{D}_{\text{I}}^{(t)}$, respectively. According to the following Theorem 7, by setting $\mathcal{Y} = \overline{\boldsymbol{Y}}_n - \boldsymbol{D}_{\text{E}}\boldsymbol{A}_{\text{E}}^{(t+1)}, \mathcal{E} = \boldsymbol{D}_{\text{E}}, \mathcal{D} = \boldsymbol{D}_{\text{I}}, \mathcal{A} = \boldsymbol{A}_{\text{I}}$, the problem (3.9) has a closed-form solution $\boldsymbol{D}_{\text{I}}^{(t+1)} = \boldsymbol{U}_{\text{I}}\boldsymbol{V}_{\text{I}}^\top$, where $\boldsymbol{U}_{\text{I}} \in \mathbb{R}^{3p^2\times(3p^2-r)}$ and $\boldsymbol{V}_{\text{I}} \in \mathbb{R}^{(3p^2-r)\times(3p^2-r)}$ are the orthogonal matrices obtained by the following SVD [**eckart1936approximation**]

$$(\boldsymbol{I} - \boldsymbol{D}_{\text{E}}\boldsymbol{D}_{\text{E}}^\top)\boldsymbol{Y}(\boldsymbol{A}_{\text{I}}^{(t+1)})^\top = \boldsymbol{U}_{\text{I}}\boldsymbol{S}_{\text{I}}\boldsymbol{V}_{\text{I}}^\top. \tag{3.10}$$

The orthogonality of internal sub-dictionary $\boldsymbol{D}_{\text{I}}^{(t+1)}$ can be checked by $(\boldsymbol{D}_{\text{I}}^{(t+1)})^\top(\boldsymbol{D}_{\text{I}}^{(t+1)}) = \boldsymbol{V}_{\text{I}}\boldsymbol{U}_{\text{I}}^\top\boldsymbol{U}_{\text{I}}\boldsymbol{V}_{\text{I}}^\top = \boldsymbol{I}_{(3p^2-r)}$. In fact, the Theorem 7 provides a sufficient and necessary condition to guarantee the existence of the closed-form solution for the internal sub-dictionary of the problem (3.9).

Theorem 1. *Let $\mathcal{A} \in \mathbb{R}^{(3p^2-r)\times M}$, $\mathcal{Y} \in \mathbb{R}^{3p^2\times M}$ be two given data matrices. $\mathcal{E} \in \mathbb{R}^{3p^2\times r}$ is a given matrix satisfying $\mathcal{E}^\top\mathcal{E} = \boldsymbol{I}_{r\times r}$, then $\hat{\mathcal{D}} = \mathcal{U}\mathcal{V}^\top$ is the necessary condition of*

$$\begin{aligned}\hat{\mathcal{D}} &= \arg\min_{\mathcal{D}} \|\mathcal{Y} - \mathcal{D}\mathcal{A}\|_F^2 \\ &s.t. \quad \mathcal{D}^\top\mathcal{D} = \boldsymbol{I}_{(3p^2-r)\times(3p^2-r)}, \mathcal{E}^\top\mathcal{D} = \boldsymbol{0}_{r\times(3p^2-r)},\end{aligned} \tag{3.11}$$

*where $\mathcal{U} \in \mathbb{R}^{3p^2\times(3p^2-r)}$ and $\mathcal{V} \in \mathbb{R}^{(3p^2-r)\times(3p^2-r)}$ are the orthogonal matrices obtained by performing economy (a.k.a. reduced) SVD [**eckart1936approximation**]:*

$$(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top = \mathcal{U}\Sigma\mathcal{V}^\top \tag{3.12}$$

Besides, if $rank(\Sigma) = 3p^2 - r$, $\hat{\mathcal{D}} = \mathcal{U}\mathcal{V}^\top$ is also the sufficient condition of problem (3.11).

The proof of the Theorem 7 can be found in Appendix B. Though the problem (3.9) has a closed-form solution by SVD [**eckart1936approximation**], the uniqueness of solution cannot be guaranteed since the matrices $(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top$ as well as $\mathcal{U}$ and $\mathcal{V}$ may be reduced to matrices of lower rank. Hence, we also analyze the uniqueness of the solution $\hat{\mathcal{D}}$ by the following Theorem 8, whose proof can be found in Appendix C.

Theorem 2. *(a) If $(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top \in \mathbb{R}^{3p^2\times(3p^2-r)}$ is nonsingular, i.e., $rank(\Sigma) = 3p^2 - r$, then the solution of $\hat{\mathcal{D}} = \mathcal{U}\mathcal{V}^\top$ is unique; (b) If $(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top$ is singular, i.e., $0 \le rank(\boldsymbol{\Sigma}) < 3p^2 - r$, then the number of possible solutions of $\hat{\mathcal{D}}$ is $2^{3p^2-r-rank(\boldsymbol{\Sigma})}$ for fixed $\mathcal{U}$ and $\mathcal{V}$.*

The above alternative updating steps are repeated until the number of iterations exceeds a preset threshold. In each step, the energy value of the objective function (3.5) is decreased and we empirically found that the proposed model usually converges in 10 iterations. We summarize the procedures in Algorithm 1.

3.3.3 The Denoising Algorithm

The denoising of the given noisy image $\mathbf{y}$ can be simultaneously done with the guided internal sub-dictionary learning process. Once we obtain the solutions of sparse coding vectors $\{\hat{\boldsymbol{\alpha}}_{n,m}^{(T)}\}$ in Eq. (3.8) and the orthogonal dictionary $\boldsymbol{D}^{(T)} = [\boldsymbol{D}_{\mathrm{E}}\ \boldsymbol{D}_{\mathrm{I}}^{(T)}]$ in Eq. (3.9), the latent clean patch $\hat{\mathbf{y}}_{n,m}$ of the m-th noisy patch in PG $\boldsymbol{Y}_n$ is reconstructed as

Algorithm 1: External Prior Guided Internal Prior Learning

Input: Matrices $\overline{\boldsymbol{Y}}_n$, external sub-dictionary $\boldsymbol{D}_{\mathrm{E}}$, parameter vector $\boldsymbol{\lambda}$
Initialization: initialize $\boldsymbol{D}^{(0)} = \boldsymbol{U}_k$ by Eq. (3.2);
for $t = 0, 1, ..., T-1$ **do**
1. Update $\boldsymbol{\alpha}_{n,m}^{(t+1)}$ by Eq. (3.7);
2. Update $\boldsymbol{D}_{\mathrm{I}}^{(t+1)}$ by Eq. (3.9);
end for
Output: Internal orthogonal dictionary $\boldsymbol{D}_{\mathrm{I}}^{(T)}$ and sparse codes $\mathbf{A}^{(T)}$.

$$\hat{\mathbf{y}}_{n,m} = \boldsymbol{D}^{(T)}\hat{\boldsymbol{\alpha}}_{n,m}^{(T)} + \boldsymbol{\mu}_n, \tag{3.13}$$

where $\boldsymbol{\mu}_n$ is the group mean of $\boldsymbol{Y}_n$. The latent clean image is then reconstructed by aggregating all the reconstructed patches in all PGs. We perform the above denoising procedures for several iterations for better denoising outputs. The proposed denoising algorithm is summarized in Algorithm 2.

Algorithm 2: External Prior Guided Internal Prior Learning for Real Noisy Image Denoising

Input: Noisy image $\mathbf{y}$, external PG prior GMM model
Initialization: $\hat{\mathbf{x}}^{(0)} = \mathbf{y}$;
for $Ite = 1 : IteNum$ **do**
1. Extracting internal PGs $\{\boldsymbol{Y}_n\}_{n=1}^{N}$ from $\hat{\mathbf{x}}^{(Ite-1)}$;
Guided Internal Subspace Clustering:
  for each PG $\boldsymbol{Y}_n$ **do**
2. Calculate group mean $\boldsymbol{\mu}_n$ and form mean subtracted PG $\overline{\boldsymbol{Y}}_n$;
3. Subspace clustering via Eq. (3.3);
  end for
Guided Internal Orthogonal Dictionary Learning:
  for the PGs in each subspace **do**
4. External PG prior guided internal orthogonal dictionary learning by solving Eq. (3.5);
5. Recover each patch in all PGs via Eq. (3.13);
  end for
6. Aggregate the recovered PGs of all subspaces to form the recovered image $\hat{\mathbf{x}}^{(Ite)}$;
end for
Output: The denoised image $\hat{\mathbf{x}}$.

3.4 Experiments

3.4.1 Implementation Details

Our proposed method has two main stages: the external prior learning stage and the external prior guided internal prior learning stage. In the first stage, we set $p = 6$ (the patch size), $M = 10$ (the number of similar patches in a PG), $W = 31$ (the window size for PG searching) and $K = 32$ (the number of Gaussian components in GMM). We learn the external GMM prior with 3.6 million PGs extracted from the Kodak PhotoCD Dataset (`http://r0k.us/graphics/kodak/`), which includes 24 high quality color images.

In the second stage, we set $r = 54$ (the number of atoms in the external sub-dictionaries); that is, we let the external sub-dictionaries have the same number of atoms as the internal sub-dictionaries to be learned. Our experiments show that setting r between 27 and 81 will lead to very similar results. For other parameters, we set $\lambda = 0.001$ (the sparse regularization parameter), $T = 2$ (the number of iterations for solving problem (3.5)), and $IteNum = 4$ (the number of iterations for Alg. 2). All parameters of our method are fixed to all experiments, which are run under the Matlab2014b environment on a machine with Intel(R) Core(TM) i7-5930K CPU of 3.5GHz and 32GB RAM.

3.4.2 The Testing Datasets

We evaluate the proposed method on three real noisy image datasets, where the images were captured under indoor or outdoor lighting conditions by different types of cameras and camera settings.

Dataset 1. The first dataset is provided in [**ncwebsite**], which includes 20 real noisy images collected under uncontrolled outdoor environment. Fig. 5.3 shows some sample images of this dataset. Since there is no “ground truth" of the noisy images, the objective measures such as PSNR cannot be computed on this dataset.

Dataset 2. The second dataset is provided in [**crosschannel2016**], which includes noisy images of 11 static scenes. The noisy images were collected under controlled indoor environment. Each scene was shot 500 times under the same camera and camera setting. The mean image of the 500 shots is roughly taken as the “ground truth", with which the PSNR can be computed.

Fig. 3.3: Some samples cropped from real noisy images of Dataset 1 [**ncwebsite**].

Since the image size is very large (about 7000×5000) and the 11 scenes share repetitive contents, the authors of [**crosschannel2016**] cropped 15 smaller images (of size 512×512) to perform experiments. In order to evaluate the proposed methods more comprehensively, we cropped 60 images of size 500×500 from the dataset for experiments. Some samples are shown in Fig. 4.4. Note that our cropped 60 images and the 15 cropped images by the authors of [**crosschannel2016**] are from different shots.

Dataset 3. The scenes of dataset 2 are mostly printed photos, and they cannot represent realistic objects and scenes with different reflectance properties. To remedy the limitation of dataset 2, we construct another dataset which contains images of 10 different scenes captured by Canon 80D and Sony A7II cameras under more ISO settings. The ISO settings in our dataset are 800, 1600, 3200, 6400, 12800 while those of dataset 2 are 1600, 3200, 6400. Similar to dataset 2, each scene was captured 500 shots, and the mean image of these 500 shots can be used a kind of ground-truth to evaluate the denoising algorithms. Fig. 3.5 shows some cropped images of the scenes in our dataset. One can see that the images contain a lot of different realistic objects with varying colors, shapes, materials, etc.

Our dataset provides real noisy images of realistic objects with different ISO settings. It can be used to more fairly evaluate the performance of different real noisy image denoising methods. Consider that the image resolution is very high (about 4000×4000), for the convenience of experimental studies, we cropped 100 (10 for each scene) smaller images (of size 512×512) from it to perform experiments. The whole dataset will be made publically available with the publication of this paper.

Fig. 3.4: Some samples cropped from real noisy images of Dataset 2 [**crosschannel2016**].

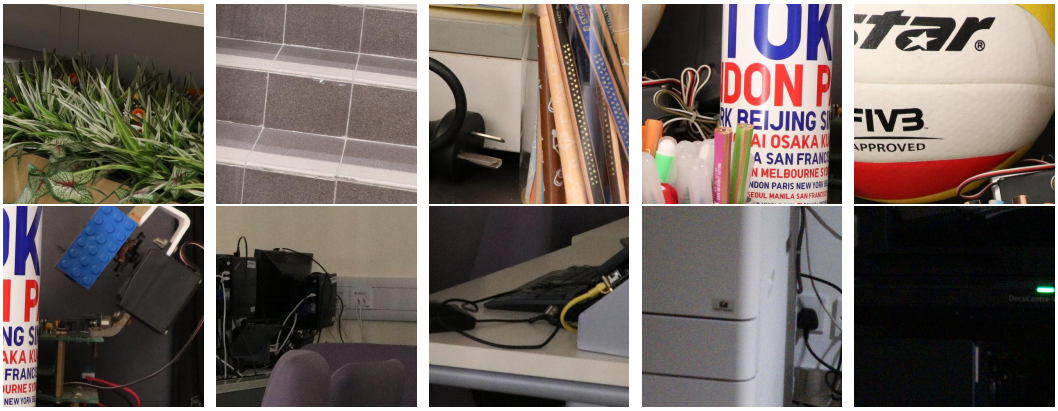
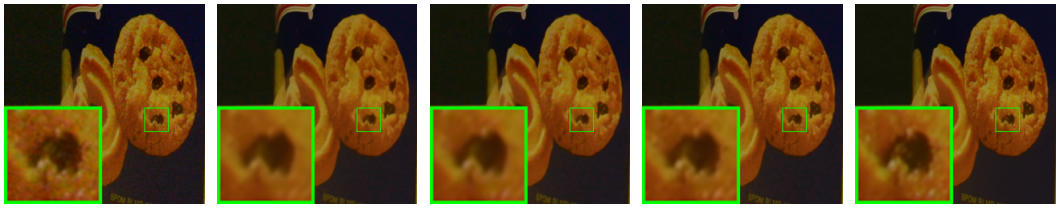


Fig. 3.5: Some samples cropped from our dataset (Dataset 3).

(a)Noisy 35.89 (b)External 39.05 (c)Internal 38.75 (d)Guided Internal **39.39** (e)Mean Image

Fig. 3.6: Denoised images and PSNR(dB) results of a region cropped from the real noisy image "Nikon D600 ISO 3200 C1" [**crosschannel2016**] by different methods. The images are better to be zoomed in on screen.



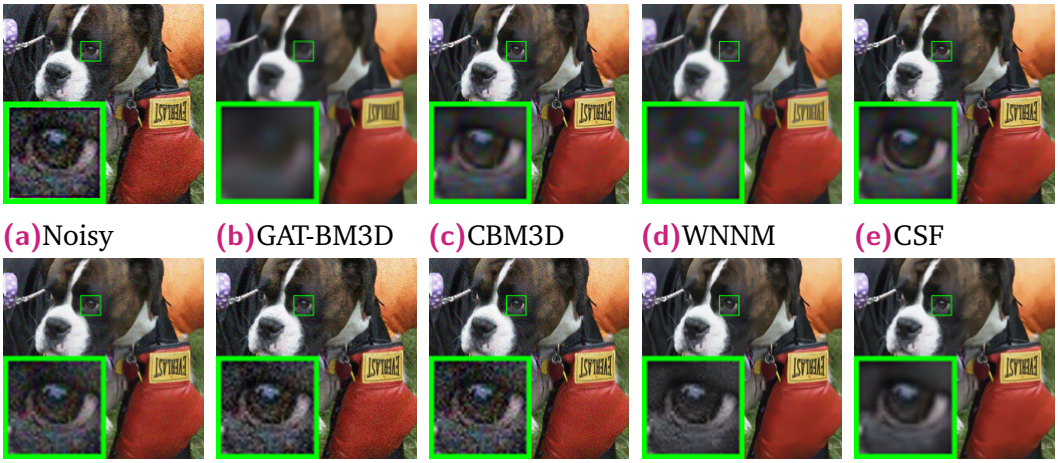
(a)Noisy 33.77 (b)External 36.97 (c)Internal 37.40 (d)Guided Internal **38.01** (e)Mean Image

Fig. 3.7: Denoised images and PSNR(dB) results of a region cropped from the real noisy image “Nikon D600 ISO 3200 C1” [**crosschannel2016**] by different methods. The images are better to be zoomed in on screen.

3.4.3 Comparison among external, internal and guided internal priors

To demonstrate the advantages of external prior guided internal prior learning, we perform real noisy image denoising by using external priors only (denoted by “External"), internal priors only (denoted by “Internal"), and the proposed guided internal priors (denoted by “Guided Internal"), respectively. For the “External" method, we utilize the full external dictionaries (i.e., $r = 108$ in Eq. (3.5)) for denoising. For the “Internal" method, the overall framework is similar to the method of [**ncsr**]. A GMM model (with $K = 32$ Gaussians) is directly learned from the PGs extracted from the given noisy image without using any external data, and then the internal orthogonal dictionaries are obtained via Eq. (3.2) to perform denoising. All parameters of the “External" and “Internal" methods are tuned to achieve their best performance.

We compare the three methods on the 60 cropped images from [**crosschannel2016**]. The average PSNR and run time are listed in Table 3.1. The best results are highlighted in bold. It can be seen that “Guided Internal" method achieves better PSNR than both “External" and “Internal" methods. In addition, the “Internal" method is very slow because it involves online GMM learning, while the “Guided Internal" method is only a little slower than the “External" method. Figs. 3.6 and 3.7 show the denoised images of two noisy images by the three methods. One can see that the “External" method is good at recovering large-scale structures (see Fig. 3.6) while the “Internal" method is good at recovering fine-scale textures (see Fig. 3.7). By utilizing external priors



(a)Noisy (b)GAT-BM3D (c)CBM3D (d)WNNM (e)CSF

(f)TNRD (g)DnCNN (h)NI (i)NC (j)Ours

Fig. 3.8: Denoised images of the real noisy image "Dog" [**ncwebsite**] by different methods. The images are better to be zoomed in on screen.

to guide the internal prior learning, our proposed method can effectively recover both the large-scale structures and fine-scale textures.

3.4.4 Comparison with State-of-the-Art Denoising Methods

Comparison methods. We compare the proposed method with state-of-the-art image denoising methods, including GAT-BM3D [**Makitalo2013Optimal**], CBM3D [**cbm3d**], WNNM [**wnnm**], MLP [**mlp**], DnCNN [**zhang2017beyond**], CSF [**csf**], TNRD [**chen2015learning**], Noise Clinic (NC) [**noiseclinic**; **ncwebsite**], Cross-Channel (CC) [**crosschannel2016**], and Neat Image (NI) [**neatimage**]. Among these methods, GAT-BM3D [**Makitalo2013Optimal**] is a state-of-the-art Poisson noise reduction method. The method CBM3D [**cbm3d**] is a state-of-the-art method for color image denoising and the noise on color images is assumed to be additive white Gaussian. The methods of WNNM, MLP, DnCNN, CSF, and TNRD are state-of-the-art Gaussian noise removal methods for grayscale images, and we apply them to each channel of color images for denoising. NC is a blind image denoising method, and NI is a set of commercial software for image denoising, which has been embedded into Photoshop and Corel PaintShop. The code of CC is not released but its results on the 15 cropped images are available at [**crosschannel2016**]. Therefore, we only compare with it on the 15 cropped images in dataset 2 from [**crosschannel2016**].

Table 3.1: Average PSNR (dB) and Run Time (seconds) of the “External", “Internal", and “Guided Internal" methods on 60 real noisy images (of size $500 \times 500 \times 3$) cropped from [**crosschannel2016**].

	Noisy	**External**	**Internal**	**Guided Internal**
PSNR	34.51	38.21	38.07	**38.75**
Time		**21.19**	312.67	22.26

Noise level of comparison methods. For the CBM3D method, the standard deviation of noise on color images should be given as a parameter. For methods of WNNM, MLP, CSF, and TNRD, the noise level in each color channel should be input. For the DnCNN method, it is trained to deal with noise in a range of levels $0 \sim 55$. We retrain the models of discriminative denoising methods MLP, CSF, and TNRD (using the released codes by the authors) at different noise levels from $\sigma = 5$ to $\sigma = 50$ with a gap of 5. The denoising is performed by processing each channel with the model trained at the same (or nearest) noise level. The noise levels $(\sigma_r, \sigma_g, \sigma_b)$ in R, G ,B channels are assumed to be Gaussian and can be estimated via some noise estimation methods [**noiselevel**; **Chen2015ICCV**]. In this paper, we employ the method [**Chen2015ICCV**] to estimate the noise level for each color channel.

Results on dataset 1. Since there is no “ground truth" for the real noisy images in dataset 1 [**ncwebsite**], we only compare the visual quality of the denoised images by different methods. (Note that method CC [**crosschannel2016**] is not compared since its code is not available. The result of MLP is not shown here due to the limit of space.) Fig. 3.8 show the denoised images of “Dog”. It can be seen that CBM3D and WNNM tend to over-smooth much the image while remaining some noise caused color artifacts. DnCNN and TNRD are likely to remain many noise-caused color artifacts across the whole image. These results demonstrate that the methods designed with Gaussian noise model are not effective for real noise removal. Though NC and NI methods are specifically developed for real noisy images, their performance on noise removal is not very satisfactory. In comparison, our proposed method recovers much better the structures and textures (such as the eye area in “Dog”) than the other competing methods. More visual comparisons can be found in the supplementary file.

Results on dataset 2. As described in section 4.2, there is a mean image for each of the 11 scenes used in dataset 2 [**crosschannel2016**], and those mean images can be roughly taken as “ground truth" images for quantitative

Table 3.2: PSNR(dB) results of different methods on 15 cropped real noisy images used in [**crosschannel2016**].

Camera Settings	**GAT-BM3D**	**CBM3D**	**WNNM**	**MLP**	**CSF**	**TNRD**	**DnCNN**	**NI**	**NC**
Canon 5D Mark III ISO = 3200	31.23	39.76	37.51	39.00	35.68	39.51	37.26	37.68	38.76
	30.55	36.40	33.86	36.34	34.03	36.47	34.13	34.87	35.69
	27.74	36.37	31.43	36.33	32.63	**36.45**	34.09	34.77	35.54
Nikon D600 ISO = 3200	28.55	34.18	33.46	34.70	31.78	34.79	33.62	34.12	**35.57**
	32.01	35.07	36.09	36.20	35.16	36.37	34.48	35.36	**36.70**
	39.78	37.13	39.86	39.33	39.98	39.49	35.41	38.68	39.28
Nikon D800 ISO = 1600	32.24	36.81	36.35	37.95	34.84	38.11	35.79	37.34	38.01
	33.86	37.76	39.99	40.23	38.42	40.52	36.08	38.57	39.05
	33.90	37.51	37.15	37.94	35.79	38.17	35.48	37.87	38.20
Nikon D800 ISO = 3200	36.49	35.05	38.60	37.55	38.36	37.69	34.08	36.95	38.07
	32.91	34.07	36.04	35.91	35.53	35.90	33.70	35.09	35.72
	40.20	34.42	39.73	38.15	40.05	38.21	33.31	36.91	36.76
Nikon D800 ISO = 6400	29.84	31.13	33.29	32.69	34.08	32.81	29.83	31.28	33.49
	27.94	31.22	31.16	32.33	32.13	32.33	30.55	31.38	32.79
	29.15	30.97	31.98	32.29	31.52	32.29	30.09	31.40	32.86
Average	32.43	35.19	35.77	36.46	35.33	36.61	33.86	35.49	36.43

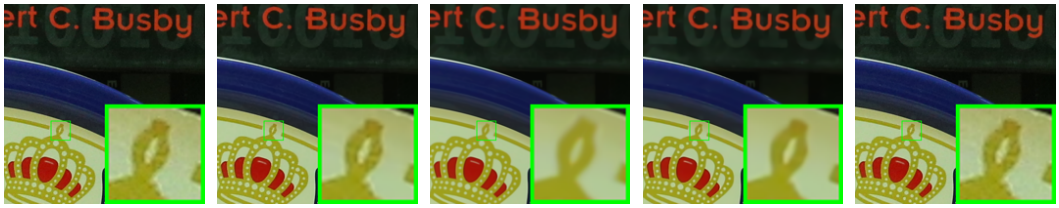
evaluation of denoising algorithms. We firstly perform quantitative comparison on the 15 cropped images used in [**crosschannel2016**]. The PSNR results of GAT-BM3D, CBM3D, WNNM, MLP, CSF, TNRD, DnCNN, NC, NI and CC are listed in Table II (The results of CC are copied from the original paper [**crosschannel2016**]). The best PSNR results of each image are highlighted in bold. One can see that on 8 out of the 15 images, our method achieves the best PSNR values. CC achieves the best PSNR on 3 of the 15 images. It should be noted that in the CC method, a specific model is trained for each camera and camera setting, while our method uses the same model for all images. On average, our proposed method has 0.27dB PSNR improvements over the second best method CC and much higher PSNR gains over other competing methods. The method GAT-BM3D does not work well on most images. This is because real world noise is much more complex than Poisson.

Fig. 3.9 shows the denoised images of one scene captured by Canon 5D Mark 3 at ISO = 3200. We can see that GAT-BM3D, CBM3D, WNNM, DnCNN, NC, NI and CC would either remain noise or generate artifacts, while TNRD over-smooths much the image. By using the external prior guided internal priors, our proposed method preserves edges and textures better than other methods, leading to visually pleasant outputs. More comparisons on visual quality and SSIM [**ssim**] index can be found in the supplementary file.

We then perform denoising experiments on the 60 images we cropped from [**crosschannel2016**]. The average PSNR results are listed in Table III (CC is not compared since the code is not available). Again, our proposed method achieves much better PSNR results than the other methods. The improvements of our method over the second best method (TNRD) are 0.43dB on PSNR. Fig. 3.10 shows the denoised images of one scene captured by Nikon D800 at ISO = 3200. We can see again that the proposed method obtain better visual quality than other competing methods. More comparisons on visual quality and SSIM can be found in the supplementary file.

Results on dataset 3. Similar to dataset 2 [**crosschannel2016**], there is a "ground truth" image for each of the 10 scenes used in our constructed dataset 3. We perform quantitative comparison on the 100 cropped images. The average PSNR results of competing methods are listed in Table IV. We can see that our proposed method achieves much better PSNR results than the other methods. The improvements of our method over the second best method (TNRD) is 0.16dB on PSNR. Fig. 3.11 shows the denoised images of one scene captured by Canon 80D at ISO = 12800. We can see again that the proposed method removes the noise while maintains better details (such as the vertical black shadow area) than other competing methods. More comparisons on visual quality and SSIM can be found in the supplementary file.

Comparison on speed. Efficiency is an important aspect to evaluate the efficiency of algorithms. We compare the speed of all competing methods except for CC. All experiments are run under the Matlab2014b environment on a machine with Intel(R) Core(TM) i7-5930K CPU of 3.5GHz and 32GB RAM. The average running time (second) of the compared methods on the 100 real noisy images is shown in Table V. The least average running time are highlighted in bold. One can easily see that the commercial software Neat Image (NI) is the fastest method with highly optimized code. For a 512×512 image, NI costs about 0.6 second. The other methods cost from 5.2 (TNRD) to 152.2 (WNNM) seconds, while the proposed method costs about 24.1 seconds. It should be noted that GAT-BM3D, CBM3D, TNRD, and NC are implemented with compiled C++ mex-function and with parallelization, while WNNM, MLP, CSF, DnCNN, and the proposed method are implemented purely in Matlab.



(a)Noisy 37.00 (b)CBM3D39.76 (c)WNNM 37.51 (d)TNRD 39.51 (e)DnCNN 37.26

(f)NI 37.68 (g)NC 38.76 (h)CC 38.37 (i)Ours **40.50** (j)Mean Image

Fig. 3.9: Denoised images and PSNR(dB) results of a region cropped from the real noisy image "Canon 5D Mark 3 ISO 3200 1" [**crosschannel2016**] by different methods. The images are better to be zoomed in on screen.

Table 3.3: Average PSNR(dB) results of different methods on 60 real noisy images cropped from [**crosschannel2016**].

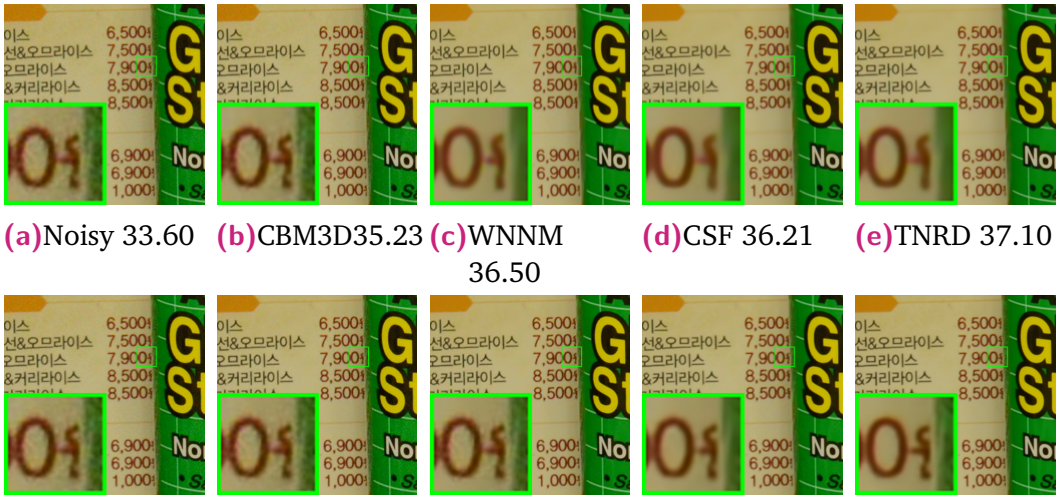
Methods	**GAT-BM3D**	**CBM3D**	**WNNM**	**MLP**	**CSF**	**TNRD**	**DnCNN**	**NI**	**NC**	**Ours**
PSNR	34.33	36.34	37.67	38.13	37.40	38.32	34.99	36.53	37.57	**38.75**

Table 3.4: Average PSNR(dB) results of different methods on 100 real noisy images cropped from our new dataset.

Methods	**GAT-BM3D**	**CBM3D**	**WNNM**	**MLP**	**CSF**	**TNRD**	**DnCNN**	**NI**	**NC**	**Ours**
PSNR	33.54	37.14	35.18	37.34	37.07	37.48	34.74	35.70	36.76	**37.64**

Table 3.5: Average Speed (sec.) results of different methods on 100 real noisy images cropped from our new dataset.

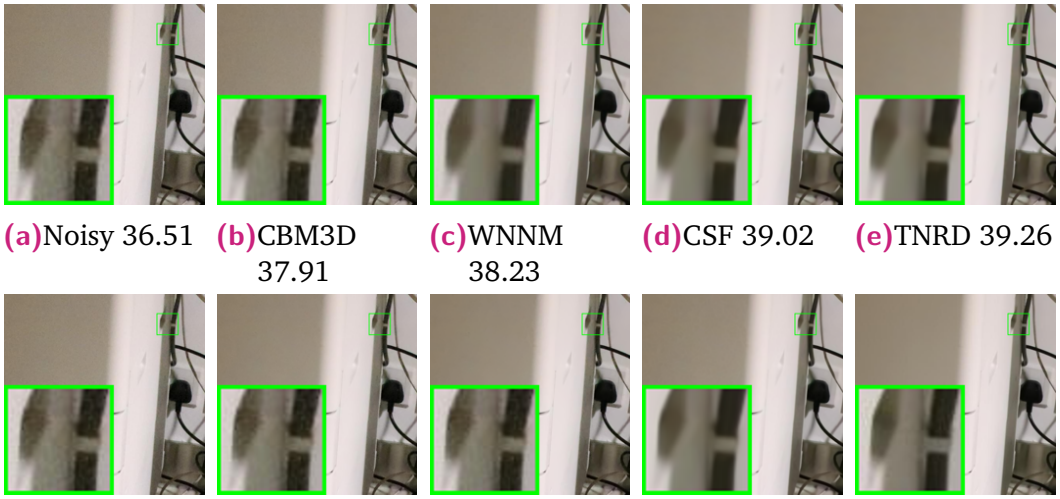
Methods	**GAT-BM3D**	**CBM3D**	**WNNM**	**MLP**	**CSF**	**TNRD**	**DnCNN**	**NI**	**NC**	**Ours**
Time	11.1	6.9	152.2	17.1	19.5	5.2	79.5	**0.6**	15.6	24.1



(a)Noisy 33.60 (b)CBM3D35.23 (c)WNNM 36.50 (d)CSF 36.21 (e)TNRD 37.10

(f)DnCNN 34.43 (g)NI 35.02 (h)NC 36.07 (i)Ours **37.50** (j)Mean Image

Fig. 3.10: Denoised images and PSNR(dB) results of a region cropped from the real noisy image "Nikon D800 ISO 3200 A3" [**crosschannel2016**] by different methods. The images are better to be zoomed in on screen.



(a)Noisy 36.51 (b)CBM3D 37.91 (c)WNNM 38.23 (d)CSF 39.02 (e)TNRD 39.26

(f)DnCNN 36.52 (g)NI 37.52 (h)NC 37.53 (i)Ours **39.41** (j)Mean Image

Fig. 3.11: Denoised images and PSNR(dB) results of a region cropped from the real noisy image "Canon 80D ISO 12800 IMG 2321" in our new dataset by different methods.. The images are better to be zoomed in on screen.

3.5 Conclusion

We proposed a new prior learning method for the real noisy image denoising problem by exploiting the useful information in both external and internal data. We first learned Gaussian Mixture Models (GMMs) from a set of clean external images as general image prior, and then employed the learned GMM model to guide the learning of adaptive internal prior from the given noisy image. Finally, a set of orthogonal dictionaries were output as the external-internal hybrid prior models for image denoising. Extensive experiments on three real noisy image datasets, including a new dataset constructed by us by different types of cameras and camera settings, demonstrated that our proposed method achieves much better performance than state-of-the-art image denoising methods in terms of both quantitative measure and perceptual quality.

4 Internal Nonlocal Self-Similarity Prior for Real Color Image Denoising: A Low Rank based Method

Image denoising is a classical yet fundamental problem for image quality enhancement in computer vision and photography systems. Most of existing denoising algorithms are designed for grayscale images, aiming to recover the clean image $\mathbf{x}$ from its noisy observation $\mathbf{y} = \mathbf{x} + \mathbf{n}$, where $\mathbf{n}$ is generally assumed to be additive white Gaussian noise (AWGN). State-of-the-art image denoising methods include sparse representation [**bm3d**], dictionary learning [**ksvd**], low-rank approximation [**wnnm**], non-local self-similarity (NSS) [**nlm**] based methods, and the combination of those techniques [**ksvd**; **bm3d**; **lssc**; **epll**; **ncsr**; **pgpd**; **wnnm**]. Recently, some discriminative denoising methods have also been developed by learning discriminative priors from pairs of clean and noisy images [**mlp**; **csf**; **chen2015learning**; **dncnn**].

When the input is a noisy RGB color image, there are mainly three strategies for color image denoising. (1) The first strategy is to apply the grayscale image denoising algorithm to each channel. However, such a straightforward solution will not exploit the spectral correlation among RGB channels, and the denoising performance may not be very satisfying. (2) The second strategy is to transform the RGB image into a less correlated color space, such as YCbCr, and perform denoising in each channel of the transformed space [**foe**; **cbm3d**]. One representative work along this line is the CBM3D algorithm [**cbm3d**]. However, the color transform will complicate the noise distribution, and the correlation among color channels is not fully exploited. (3) The third strategy is to perform joint denoising on the RGB channels simultaneously for better use of the spectral correlation. For example, the patches from RGB channels are concatenated as a long vector for processing [**mairal2008sparse**; **Zhu_2016_CVPR**].

Though joint denoising of RGB channels is a more promising way for color image denoising, it is not a trivial extension from single channel (grayscale im-

age) to multiple channels (color image). The noise in standard RGB (sRGB) space can be approximately modeled as AWGN, but it has different variances for different channels [**Liu2008**; **Leungtip**; **crosschannel2016**] due to the sensor characteristics and on-board processing steps in digital camera pipelines [**crosschannel2016**; **karaimer_brown_ECCV_2016**]. This makes the real color image denoising problem much more complex. If the three channels are treated equally in the joint denoising process, false colors or artifacts can be generated [**mairal2008sparse**]. How to account for the different noise characteristics in color channels, and how to effectively exploit the within and cross channel correlation are the key issues for designing a good color image denoising method.

This paper presents a new color image denoising algorithm. Considering that the weighted nuclear norm minimization (WNNM) method [**wnnm**; **wnnmijcv**], which exploits the image NSS property via low rank regularization, has achieved excellent denoising performance on grayscale images, we propose to extend WNNM to real color image denoising. More specifically, we propose a multi-channel WNNM (MC-WNNM) model, which concatenates the patches from RGB channels for rank minimization but introduces a weight matrix to adjust the contributions of the three channels based on their noise levels. The proposed MC-WNNM model no longer has a closed-form solution as in the original WNNM model [**wnnmijcv**]. We reformulate it into a linear equality-constrained problem with two variables, and solve the relaxed problem under the alternating direction method of multipliers [**admm**] framework. Each variable can be updated with closed-form solutions, and the convergence analysis is given to guarantee a rational termination of the proposed algorithm.

4.1 Related Work

4.1.1 Weighted Nuclear Norm Minimization

As a generalization to the nuclear norm minimization (NNM) model [**cai2010singular**], the weighted nuclear norm minimization (WNNM) model [**wnnm**; **wnnmijcv**] is described as

$$\min_{\mathbf{X}} \|\mathbf{Y}-\mathbf{X}\|_F^2 + \|\mathbf{X}\|_{\boldsymbol{w},*}, \tag{4.1}$$

where $\|\mathbf{X}\|_{\boldsymbol{w},*} = \sum_i w_i \sigma_i(\mathbf{X})$ is the weighted nuclear norm of matrix $\mathbf{X}$, $\boldsymbol{w} = [w_1, ..., w_n]^\top$ ($w_i \geq 0$) is the weight vector, and $\sigma_i(\mathbf{X})$ is the ith singular value of $\mathbf{X}$. According to the Corollary 1 of [**wnnmijcv**], if the weights are non-decreasing, the problem (5.1) has a closed-form solution:

$$\hat{\mathbf{X}} = \mathbf{U}\mathcal{S}_{\boldsymbol{w}/2}(\mathbf{\Sigma})\mathbf{V}^\top, \tag{4.2}$$

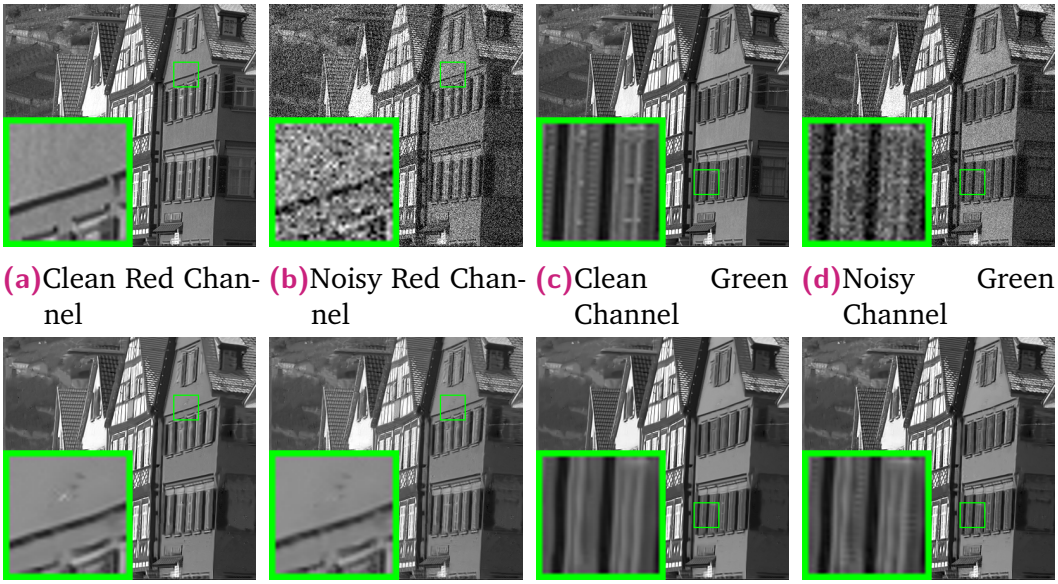
where $\mathbf{Y} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ is the singular value decomposition (SVD) [**eckart1936approximation**] of $\mathbf{Y}$ and $\mathcal{S}_{\boldsymbol{w}/2}(\bullet)$ is the generalized soft-thresholding operator with weight vector $\boldsymbol{w}$:

$$\mathcal{S}_{\boldsymbol{w}/2}(\mathbf{\Sigma}_{ii}) = \max(\mathbf{\Sigma}_{ii} - w_i/2, 0). \tag{4.3}$$

WNNM has demonstrated highly competitive denoising performance on grayscale images. However, if we directly extend it to color image denoising by concatenating the patches from RGB channels, denoising artifacts may happen (please refer to Fig. 5.1 and the section of experimental results). In this paper, we propose a multi-channel WNNM (MC-WNNM) model for color image denoising, which preserves the power of WNNM and is able to address the noise differences among different channels.

4.1.2 Real Color Image Denoising

During the last decade, a few methods have been proposed for real color image denoising. Among them, the CBM3D method [**cbm3d**] is a representative one, which first transforms the RGB image into a luminance-chrominance space (e.g., YCbCr) and then applies the benchmark BM3D method [**bm3d**] to each channel separately. The non-local similar patches are grouped by the luminance channel. In [**Liu2008**], Liu et al. proposed the "Noise Level Function" to estimate and remove the noise for each channel in natural images. However, processing each channel separately would often achieve inferior performance to processing the color channels jointly [**mairal2008sparse**]. Therefore, the methods [**noiseclinic**; **ncwebsite**; **Zhu_2016_CVPR**] perform real color image denoising by concatenating the patches of RGB channels into a long vector. However, the concatenation treats each channel equally and ignores the different noise statistics among these channels. The method in [**crosschannel2016**] models the cross-channel noise in real noisy images as multivariate Gaussian and the noise is removed by the Bayesian non-local means filter [**kervrann2007bayesian**]. The commercial software Neat Image [**neatimage**] estimates the noise parameters from a flat region



(a)Clean Red Channel (b)Noisy Red Channel (c)Clean Green Channel (d)Noisy Green Channel

(e)Denoised Red by WNNM (f)Denoised Red by MC-WNNM (g)Denoised Green by WNNM (h)Denoised Green by MC-WNNM

Fig. 4.1: The Red and Green channels of the image "kodim08" from the Kodak PhotoCD Dataset, its synthetic noisy version, and the images recovered by the concatenated WNNM and the proposed MC-WNNM methods.

of the given noisy image and filters the noise accordingly. The methods in [**crosschannel2016**; **neatimage**] ignore the non-local self-similarity of natural images [**bm3d**; **wnnm**].

In this paper, we present an effective multi-channel image denoising algorithm, which utilizes the strong low-rank prior of image non-local similar patches, and introduces a weight matrix to balance the multi-channels based on their different noise levels.

4.2 The Proposed Color Image Denoising Algorithm

4.2.1 The Multi-channel Weighted Nuclear Norm Minimization Model

The color image denoising problem is to recover the clean image $\mathbf{x}_c$ from its noisy version $\mathbf{y}_c = \mathbf{x}_c + \mathbf{n}_c$, where $c = \{r, g, b\}$ is the index of R, G, B channels and $\mathbf{n}_c$ is the noise in the c channel. Patch based image denoising [**foe**; **ksvd**; **bm3d**; **lssc**; **epll**; **ncsr**; **mlp**; **csf**; **wnnm**; **pgpd**; **chen2015learning**; **dncnn**] has achieved a great success in the last decade. Given a noisy color image $\mathbf{y}_c$, each local patch of size $p \times p \times 3$ is extracted and stretched to a patch vector, denoted by $\mathbf{y} = [\mathbf{y}_r^\top \ \mathbf{y}_g^\top \ \mathbf{y}_b^\top]^\top \in \mathbb{R}^{3p^2}$, where $\mathbf{y}_r, \mathbf{y}_g, \mathbf{y}_b \in \mathbb{R}^{p^2}$ are the corresponding patches in R, G, B channels. For each local patch $\mathbf{y}$, we search the M most similar patches to it (including $\mathbf{y}$ itself) by Euclidean distance in a relatively large local window around it. By stacking the M similar patches column by column, we form a noisy patch matrix $\mathbf{Y} = \mathbf{X} + \mathbf{N} \in \mathbb{R}^{3p^2 \times M}$, where $\mathbf{X}$ and $\mathbf{N}$ are the corresponding clean and noise patch matrices.

The noise in standard RGB (sRGB) space could be approximately modeled as additive white Gaussian (AWGN), but noise in different channels has different variances [**Liu2008**; **Leungtip**; **crosschannel2016**]. Therefore, it is problematic to directly apply some grayscale denoising methods to the concatenated vectors $\mathbf{y}$ or matrices $\mathbf{Y}$. To better illustrate this point, in Fig. 5.1, we show a clean image "kodim08" (only the R and G channels are shown due to limit of space), its noisy version generated by adding AWGN to each channel, and the denoised image by applying WNNM [**wnnmijcv**] to the concatenated patch matrix $\mathbf{Y}$. The standard deviations of AWGN added to the R, G, B channels are $\sigma_r = 40$, $\sigma_g = 20$, $\sigma_b = 30$, respectively. To make WNNM applicable to color image denoising, we set the noise standard deviation as the average deviation of the whole noisy image, i.e., $\sigma = \sqrt{(\sigma_r^2 + \sigma_g^2 + \sigma_b^2)/3} \approx 31.1$. From Fig. 5.1, one can see that the concatenated WNNM remains some noise in the R channel while over-smoothing the G channel. This is because it processes R and G channels equally without considering their differences in noise corruption.

Clearly, a more effective color image denoising algorithm should consider the different noise strength in color channels. To this end, we introduce a

weight matrix $\mathbf{W}$ to balance the noise in the RGB channels, and present the following multi-channel WNNM (MC-WNNM) model:

$$\min_{\mathbf{X}} \|\mathbf{W}(\mathbf{Y}-\mathbf{X})\|_F^2 + \|\mathbf{X}\|_{\boldsymbol{w},*}. \tag{4.4}$$

We follow the method in [**wnnmijcv**] to set the weight vector $\boldsymbol{w}$ on nuclear norm as $w_i^{k+1} = C/(|\sigma_i(\mathbf{X}_k)|+\epsilon)$, where $\epsilon > 0$ is a small number to avoid zero numerator and $\sigma_i(\mathbf{X}_k)$ is the ith singular value of the estimated data matrix $\mathbf{X}$ at the kth iteration. Note that if $\sigma_r = \sigma_g = \sigma_b$, the proposed MC-WNNM model will be reduced to the concatenated WNNM model. With an appropriate setting of the weight matrix $\mathbf{W}$ and a good optimization algorithm, the proposed MC-WNNM model will lead to much better color image denoising results. As shown in Figs. 5.1(f) and 5.1(h), MC-WNNM removes clearly the noise in the R channel while preserving textures effectively in the G channel.

4.2.2 The Setting of Weight Matrix $\mathbf{W}$

Let's denote the noisy patch matrix by $\mathbf{Y} = [\mathbf{Y}_r^\top\ \mathbf{Y}_g^\top\ \mathbf{Y}_b^\top]^\top$, where $\mathbf{Y}_r, \mathbf{Y}_g, \mathbf{Y}_b$ are sub-matrices of similar patches in R, G, B channels, respectively. The corresponding clean matrix is $\mathbf{X} = [\mathbf{X}_r^\top\ \mathbf{X}_g^\top\ \mathbf{X}_b^\top]^\top$, where $\mathbf{X}_r, \mathbf{X}_g, \mathbf{X}_b$ are similarly defined. The weight matrix $\mathbf{W}$ can be determined under the *maximum a-posterior* (MAP) estimation framework:

$$\begin{aligned}\hat{\mathbf{X}} &= \arg\max_{\mathbf{X}} \ln P(\mathbf{X}|\mathbf{Y}, \boldsymbol{w}) \\ &= \arg\max_{\mathbf{X}} \{\ln P(\mathbf{Y}|\mathbf{X}) + \ln P(\mathbf{X}|\boldsymbol{w})\}.\end{aligned} \tag{4.5}$$

The log-likelihood term $\ln P(\mathbf{Y}|\mathbf{X})$ is characterized by the statistics of noise. According to [**Leungtip**], we assume that the noise is independent among RGB channels and independently and identically distributed (i.i.d.) in each channel with Gaussian distribution and standard deviations $\{\sigma_r, \sigma_g, \sigma_b\}$. There is:

$$P(\mathbf{Y}|\mathbf{X}) = \prod_{c\in\{r,g,b\}} (2\pi\sigma_c^2)^{-\frac{3p^2}{2}} \exp\left(-\frac{1}{2\sigma_c^2}\|\mathbf{Y}_c - \mathbf{X}_c\|_F^2\right). \tag{4.6}$$

For the latent data $\mathbf{X}$, the small weighted nuclear norm prior is imposed on it, i.e., $\|\mathbf{X}\|_{\boldsymbol{w},*} = \sum_i w_i\sigma_i(\mathbf{X})$ should be sparsely distributed. We let it be:

$$P(\mathbf{X}|\boldsymbol{w}) \propto \exp\left(-\frac{1}{2}\|\mathbf{X}\|_{\boldsymbol{w},*}\right). \tag{4.7}$$

Putting (5.7) and (5.6) into (5.5), we have

$$\begin{aligned}\hat{\mathbf{X}} &= \arg\min_{\mathbf{X}} \sum_{c\in\{r,g,b\}} \frac{1}{\sigma_c^2}\|(\mathbf{Y}_c - \mathbf{X}_c)\|_F^2 + \|\mathbf{X}\|_{\boldsymbol{w},*} \\ &= \arg\min_{\mathbf{X}} \|\mathbf{W}(\mathbf{Y}-\mathbf{X})\|_F^2 + \|\mathbf{X}\|_{\boldsymbol{w},*},\end{aligned} \tag{4.8}$$

with

$$\mathbf{W} = \begin{pmatrix} \sigma_r^{-1}\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \sigma_g^{-1}\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \sigma_b^{-1}\mathbf{I} \end{pmatrix}, \tag{4.9}$$

where $\mathbf{I} \in \mathbb{R}^{p^2\times p^2}$ is the identity matrix.

Clearly, the weight matrix $\mathbf{W}$ is diagonal and determined by the noise standard deviation in each channel. The stronger the noise in a channel, the less the contribution that channel should make to the estimation of $\mathbf{X}$. Our experimental results (refer to Section 4 please) on synthetic and real noisy images clearly demonstrate the advantages of MC-WNNM over WNNM and other methods in color image denoising.

4.2.3 Model Optimization

The proposed MC-WNNM model does not have an analytical solution. In the WNNM model [**wnnmijcv**], when the weights assigned on singular values are in a non-descending order, the weighted nuclear norm proximal operator can have a global optimum with closed-form solution. Unfortunately, such a property is not valid for the MC-WNNM model because a weight matrix $\mathbf{W}$ is assigned to the rows of data matrix $\mathbf{X}$. This makes the proposed model more difficult to solve than the original WNNM model.

We employ the variable splitting method [**courant1943**; **Eckstein1992**] to solve the MC-WNNM model. By introducing an augmented variable $\mathbf{Z}$, the MC-WNNM model can be reformulated as a linear equality-constrained problem with two variables $\mathbf{X}$ and $\mathbf{Z}$:

$$\min_{\mathbf{X},\mathbf{Z}} \|\mathbf{W}(\mathbf{Y}-\mathbf{X})\|_F^2 + \|\mathbf{Z}\|_{\boldsymbol{w},*} \quad \text{s.t.} \quad \mathbf{X} = \mathbf{Z}. \tag{4.10}$$

Since the objective function is separable w.r.t. the two variables, the problem (5.10) can be solved under the alternating direction method of multipliers (ADMM) [**admm**] framework. The augmented Lagrangian function is:

$$\begin{aligned}\mathcal{L}(\mathbf{X},\mathbf{Z},\mathbf{A},\rho) =& \|\mathbf{W}(\mathbf{Y}-\mathbf{X})\|_F^2 + \|\mathbf{Z}\|_{\boldsymbol{w},*} \\ &+ \langle \mathbf{A}, \mathbf{X}-\mathbf{Z}\rangle + \frac{\rho}{2}\|\mathbf{X}-\mathbf{Z}\|_F^2,\end{aligned} \tag{4.11}$$

where $\mathbf{A}$ is the augmented Lagrangian multiplier and $\rho > 0$ is the penalty parameter. We initialize the matrix variables $\mathbf{X}_0$, $\mathbf{Z}_0$, and $\mathbf{A}_0$ to be zero matrix and $\rho_0 > 0$ to be a suitable value. Denote by $(\mathbf{X}_k, \mathbf{Z}_k)$ and $\mathbf{A}_k$ the optimization variables and Lagrange multiplier at iteration k ($k = 0, 1, 2, ...$), respectively. By taking derivatives of the Lagrangian function $\mathcal{L}$ w.r.t. $\mathbf{X}$ and $\mathbf{Z}$ and setting the derivative function to be zero, we can alternatively update the variables as follows:

(1) **Update $\mathbf{X}$ while fixing $\mathbf{Z}$ and $\mathbf{A}$**:

$$\mathbf{X}_{k+1} = \arg\min_{\mathbf{X}} \|\mathbf{W}(\mathbf{Y}-\mathbf{X})\|_F^2 + \frac{\rho_k}{2}\|\mathbf{X}-\mathbf{Z}_k+\rho_k^{-1}\mathbf{A}_k\|_F^2. \tag{4.12}$$

This is a standard least squares regression problem with closed-form solution:

$$\mathbf{X}_{k+1} = (\mathbf{W}^\top\mathbf{W} + \frac{\rho_k}{2}\mathbf{I})^{-1}(\mathbf{W}^\top\mathbf{W}\mathbf{Y} + \frac{\rho_k}{2}\mathbf{Z}_k - \frac{1}{2}\mathbf{A}_k). \tag{4.13}$$

(2) **Update $\mathbf{Z}$ while fixing $\mathbf{X}$ and $\mathbf{A}$**:

$$\mathbf{Z}_{k+1} = \arg\min_{\mathbf{Z}} \frac{\rho_k}{2}\|\mathbf{Z}-(\mathbf{X}_{k+1}+\rho_k^{-1}\mathbf{A}_k)\|_F^2 + \|\mathbf{Z}\|_{\boldsymbol{w},*}. \tag{4.14}$$

According to the Theorem 1 in [**wnnmijcv**], given the SVD of $\mathbf{X}_{k+1}+\rho_k^{-1}\mathbf{A}_k$, i.e., $\mathbf{X}_{k+1}+\rho_k^{-1}\mathbf{A}_k = \mathbf{U}_k\boldsymbol{\Sigma}_k\mathbf{V}_k^\top$, where $\boldsymbol{\Sigma}_k = \begin{pmatrix}\text{diag}(\sigma_1,\sigma_2,...,\sigma_M)\\ \mathbf{0}\end{pmatrix} \in \mathbb{R}^{3p^2\times M}$ (without loss of generality, we assume that $3p^2 \geq M$), the global optimum of the above problem is $\hat{\mathbf{Z}}_{k+1} = \mathbf{U}_k\hat{\boldsymbol{\Sigma}}_k\mathbf{V}_k^\top$, where $\hat{\boldsymbol{\Sigma}}_k = \begin{pmatrix}\text{diag}(\hat{\sigma}_1,\hat{\sigma}_2,...,\hat{\sigma}_M)\\ \mathbf{0}\end{pmatrix} \in \mathbb{R}^{3p^2\times M}$ and $(\hat{\sigma}_1,\hat{\sigma}_2,...,\hat{\sigma}_M)$ is the solution to the following convex optimization problem:

$$\begin{aligned}\min_{\hat{\sigma}_1,\hat{\sigma}_2,...,\hat{\sigma}_M} & \sum_{i=1}^{M}(\sigma_i-\hat{\sigma}_i)^2 + \frac{2w_i}{\rho_k}\hat{\sigma}_i \\ \text{s.t.} \quad & \hat{\sigma}_1 \geq \hat{\sigma}_2 \geq ... \geq \hat{\sigma}_M \geq 0.\end{aligned} \tag{4.15}$$

Algorithm 1: Solve MC-WNNM via ADMM

Input: Matrices $\mathbf{Y}$ and $\mathbf{W}$, $\mu > 1$, Tol > 0, K_1;
Initialization: $\mathbf{X}_0 = \mathbf{Z}_0 = \mathbf{A}_0 = \mathbf{0}$, $\rho_0 > 0$,
T = False, $k = 0$;
While (T == false) **do**
1. Update $\mathbf{X}_{k+1}$ as
$\mathbf{X}_{k+1} = (\mathbf{W}^\top\mathbf{W} + \frac{\rho_k}{2}\mathbf{I})^{-1}(\mathbf{W}^\top\mathbf{W}\mathbf{Y} + \frac{\rho_k}{2}\mathbf{Z}_k - \frac{1}{2}\mathbf{A}_k)$
2. Update $\mathbf{Z}_{k+1}$ by solving the problem
$\min_{\mathbf{Z}} \frac{\rho_k}{2}\|\mathbf{Z} - (\mathbf{X}_{k+1} + \rho_k^{-1}\mathbf{A}_k)\|_F^2 + \|\mathbf{Z}\|_{\boldsymbol{w},*}$
3. Update $\mathbf{A}_{k+1}$ as $\mathbf{A}_{k+1} = \mathbf{A}_k + \rho_k(\mathbf{X}_{k+1} - \mathbf{Z}_{k+1})$
4. Update $\rho_{k+1} = \mu * \rho_k$;
5. $k \leftarrow k + 1$;
if (Convergence condition is satisfied) or ($k \geq K_1$)
6. T $\leftarrow$ True
end if
end while
Output: Matrices $\mathbf{X}$ and $\mathbf{Z}$.

According to the Remark 1 in [**wnnmijcv**], the problem above has closed-form solution ($i = 1, 2, ..., M$):

$$\hat{\sigma}_i = \begin{cases} 0 & \text{if } c_2 < 0 \\ \frac{c_1 + \sqrt{c_2}}{2} & \text{if } c_2 \geq 0 \end{cases}, \tag{4.16}$$

where $c_1 = \sigma_i - \epsilon$, $c_2 = (\sigma_i - \epsilon)^2 - \frac{8C}{\rho_k}$, $\epsilon > 0$ is a small number, and C is set as $\sqrt{2M}$ by experience in [**wnnmijcv**].
(3) **Update $\mathbf{A}$ while fixing $\mathbf{X}$ and $\mathbf{Z}$**:

$$\mathbf{A}_{k+1} = \mathbf{A}_k + \rho_k(\mathbf{X}_{k+1} - \mathbf{Z}_{k+1}). \tag{4.17}$$

(4) **Update** ρ_k: $\rho_{k+1} = \mu * \rho_k$, where $\mu > 1$.

The above alternative updating steps are repeated until the convergence condition is satisfied or the number of iterations exceeds a preset threshold. The convergence condition of the ADMM algorithm is: $\|\mathbf{X}_{k+1} - \mathbf{Z}_{k+1}\|_F \leq$ Tol, $\|\mathbf{X}_{k+1} - \mathbf{X}_k\|_F \leq$ Tol, and $\|\mathbf{Z}_{k+1} - \mathbf{Z}_k\|_F \leq$ Tol are simultaneously satisfied, where Tol > 0 is a small tolerance number. We summarize the updating procedures in Algorithm 1. The convergence analysis of the proposed Algorithm 1 is given in Theorem 7. Note that since the weighted nuclear norm is non-convex in general, we employ an unbounded sequence of $\{\rho_k\}$ here to make sure that Algorithm 1 converges.

Theorem 3. *Assume that the weights in $\boldsymbol{w}$ are in a non-descending order, the sequences $\{\mathbf{X}_k\}$, $\{\mathbf{Z}_k\}$, and $\{\mathbf{A}_k\}$ generated in Algorithm 1 satisfy:*

$$(a)\ \lim_{k\to\infty} \|\mathbf{X}_{k+1} - \mathbf{Z}_{k+1}\|_F = 0; \tag{4.18}$$

$$(b)\ \lim_{k\to\infty} \|\mathbf{X}_{k+1} - \mathbf{X}_k\|_F = 0; \tag{4.19}$$

$$(c)\ \lim_{k\to\infty} \|\mathbf{Z}_{k+1} - \mathbf{Z}_k\|_F = 0. \tag{4.20}$$

Proof. We give a sketch proof here and detailed proof of this theorem can be found in the supplementary file.

We first prove that the sequence $\{\mathbf{A}_k\}$ generated by Algorithm 1 is upper bounded. Since $\{\rho_k\}$ is unbounded, i.e., $\lim_{k\to\infty} \rho_k = +\infty$, we can prove that the sequence of Lagrangian function $\{\mathcal{L}(\mathbf{X}_{k+1}, \mathbf{Z}_{k+1}, \mathbf{A}_k, \rho_k)\}$ is also upper bounded. Hence, both $\{\mathbf{W}(\mathbf{Y} - \mathbf{X}_k)\}$ and $\{\mathbf{Z}_k\}$ are upper bounded. Then $\{\mathbf{X}_k\}$ is also upper bounded. According to Eq. (5.17), we can prove that $\lim_{k\to\infty} \|\mathbf{X}_{k+1} - \mathbf{Z}_{k+1}\|_F = \lim_{k\to\infty} \rho_k^{-1}\|\mathbf{A}_{k+1} - \mathbf{A}_k\|_F = 0$, and (a) is proved. Then we can prove that $\lim_{k\to\infty} \|\mathbf{X}_{k+1} - \mathbf{X}_k\|_F \le \lim_{k\to\infty}(\|(\mathbf{W}^\top\mathbf{W} + \frac{\rho_k}{2}\mathbf{I})^{-1}(\mathbf{W}^\top\mathbf{W}\mathbf{Y} - \mathbf{W}^\top\mathbf{W}\mathbf{Z}_k - \frac{1}{2}\mathbf{A}_k)\|_F + \rho_k^{-1}\|\mathbf{A}_k - \mathbf{A}_{k-1}\|_F) = 0$ and hence (b) is proved. Finally, (c) can be proved by checking that $\lim_{k\to\infty} \|\mathbf{Z}_{k+1} - \mathbf{Z}_k\| \le \lim_{k\to\infty}(\|\boldsymbol{\Sigma}_{k-1} - \mathcal{S}_{\boldsymbol{w}/\rho_{k-1}}(\boldsymbol{\Sigma}_{k-1})\|_F + \|\mathbf{X}_{k+1} - \mathbf{X}_k\|_F + \|\rho_{k-1}^{-1}\mathbf{A}_{k-1} + \rho_k^{-1}\mathbf{A}_{k+1} - \rho_k^{-1}\mathbf{A}_k\|_F) = 0$, where $\mathbf{U}_{k-1}\boldsymbol{\Sigma}_{k-1}\mathbf{V}_{k-1}^\top$ is the SVD of $\mathbf{X}_k + \rho_{k-1}^{-1}\mathbf{A}_{k-1}$. □

4.2.4 The Denoising Algorithm

Given a noisy color image $\mathbf{y}_c$, suppose that we have extracted N local patches $\{\mathbf{y}_j\}_{j=1}^N$ and their similar patches. N noisy patch matrices $\{\mathbf{Y}_j\}_{j=1}^N$ can be formed to estimate the clean matrices $\{\mathbf{X}_j\}_{j=1}^N$. The patches in matrices $\{\mathbf{X}_j\}_{j=1}^N$ are aggregated to form the denoised image $\hat{\mathbf{x}}_c$. To obtain better denoising results, we perform the above denoising procedures for several rounds. The proposed MC-WNNM based color image denoising algorithm is summarized in Algorithm 2.

4.2.5 Complexity Analysis

In Algorithm 1 for solving the MC-WNNM model via ADMM, the cost for updating $\boldsymbol{X}$ is $\mathcal{O}(\max(p^4M, M^3))$, while the cost for updating $\boldsymbol{Z}$ is $\mathcal{O}(p^4M + M^3)$. The costs for updating $\boldsymbol{A}$ and ρ can be ignored. So the overall complexity

Algorithm 2: Color Image Denoising by MC-WNNM

Input: Noisy image $\mathbf{y}_c$, noise levels $\{\sigma_r, \sigma_g, \sigma_b\}$, K_2;
Initialization: $\hat{\mathbf{x}}_c^{(0)} = \mathbf{y}_c$, $\mathbf{y}_c^{(0)} = \mathbf{y}_c$;
for $k = 1 : K_2$ **do**
1. Set $\mathbf{y}_c^{(k)} = \hat{\mathbf{x}}_c^{(k-1)}$;
2. Extract local patches $\{\mathbf{y}_j\}_{j=1}^N$ from $\mathbf{y}_c^{(k)}$;
   for each patch $\mathbf{y}_j$ **do**
3. Search non-local similar patches $\mathbf{Y}_j$;
4. Apply the MC-WNNM model (5.10) to $\mathbf{Y}_j$ and obtain the estimated $\mathbf{X}_j$;
   end for
5. Aggregate $\{\mathbf{X}_j\}_{j=1}^N$ to form the image $\hat{\mathbf{x}}_c^{(k)}$;
end for
Output: Denoised image $\hat{\mathbf{x}}_c^{(K_2)}$.

is $\mathcal{O}((p^4M + M^3)K_1)$, where K_1 is the number of iterations. In Algorithm 2 for image denoising, we consider the number of patches N extracted from the input noisy image and the number of itertations K_2 and ignore the cost for searching similar patches. The overall cost is $\mathcal{O}((p^4M + M^3)K_1K_2N)$.

4.3 Experiments

We evaluate the proposed MC-WNNM method on synthetic and real noisy color images. We compare the proposed method with state-of-the-art denoising methods, including CBM3D [**cbm3d**], MLP [**mlp**], WNNM [**wnnm**], TNRD [**chen2015learning**], DnCNN [**dncnn**] "Noise Clinic" (NC) [**noiseclinic**; **ncwebsite**], CC [**crosschannel2016**], and the commercial software Neat Image (NI) [**neatimage**]. The Matlab source code of our MC-WNNM algorithm can be downloaded at `http://www4.comp.polyu.edu.hk/~cslzhang/code/MCWNNM.zip`.

4.3.1 Experimental Settings

Noise level estimation. For most of the competing denoising algorithms, the standard deviation of noise should be given as a parameter. In synthetic experiments, the noise levels $(\sigma_r, \sigma_g, \sigma_b)$ in R, G ,B channels are assumed to be known. In the case of real noisy images, the noise levels can be estimated via some noise estimation methods [**noiselevel**; **Chen2015ICCV**]. In this

paper, we employ the method [**Chen2015ICCV**] to estimate the noise level for each color channel.

Noise level of comparison methods. For the CBM3D method [**cbm3d**], a single parameter of noise level should be input. We set the noise level as

$$\sigma = \sqrt{(\sigma_r^2 + \sigma_g^2 + \sigma_b^2)/3}. \tag{4.21}$$

The methods of MLP [**mlp**] and TNRD [**chen2015learning**] are originally designed for grayscale images. We retrain their models (using the released codes by the authors) at different noise levels from $\sigma = 5$ to $\sigma = 75$ with a gap of 5. The denoising on color images is performed by processing each channel with the model trained at the same (or nearest) noise level.

Comparison with WNNM. In order to make a full and fair comparison with the original WNNM method [**wnnmijcv**], we implement WNNM for color image denoising in three ways. 1) We apply WNNM to each color channel separately with the corresponding noise levels $\sigma_r, \sigma_g, \sigma_b$. We call this method "WNNM-1". 2) We perform WNNM on the concatenated matrix $\mathbf{Y}$ formed by the patches in RGB channels, while the input noise level σ is computed by Eq. (4.21). We call this method "WNNM-2". 3) We set the weight matrix $\mathbf{W}$ as $\mathbf{W} = \sigma^{-1}\mathbf{I}$ in the proposed MC-WNNM model, and use our developed algorithm for denoising. We call this method "WNNM-3".

For a fair comparison, we tune the parameters of WNNM-1, WNNM-2, WNNM-3 and MC-WNNM to achieve their best denoising performance. The detailed parameters are as follows: we set the patch size as $p = 6$, the number of non-local similar patches as $M = 70$, the window size for searching similar patches as 40×40. For WNNM-3 and MC-WNNM, the updating parameter is set as $\mu = 1.001$. The number of iterations in Algorithm 1 is set as $K_1 = 10$. The number of iterations K_2 in Algorithm 2 and the initial penalty parameter ρ_0 will be given in the following sub-sections.

4.3.2 Experiments on Synthetic Noisy Color Images

We first compare MC-WNNM with the competing denoising methods [**cbm3d**; **mlp**; **chen2015learning**; **dncnn**; **noiseclinic**; **neatimage**] on the 24 color images from the Kodak PhotoCD Dataset (`http://r0k.us/graphics/kodak/`). The noisy images are generated by adding AWGN to each of the R, G, B

Fig. 4.2: The 15 cropped real noisy images used in [**crosschannel2016**].

channels, respectively. In the main paper, we report the results by setting $\sigma_r = 40, \sigma_g = 20, \sigma_b = 30$. More results with other noise settings can be found in the supplementary file. For WNNM-3 and MC-WNNM, the initial penalty parameter is set as $\rho_0 = 10$ and $\rho_0 = 3$, respectively. The number of iterations in Algorithm 2 is set as $K_2 = 8$.

The PSNR results by competing methods are listed in Table 5.1, while the best PSNR result for each image is highlighted in bold. One can see that on all the 24 images, our method achieves the highest PSNR values among the competing methods. On average, MC-WNNM achieves 0.47dB, 0.48dB and 1.09dB improvements over WNNM-1, WNNM-2 and WNNM-3, respectively. For space limitation, we leave the visual comparisons of the synthetic noisy image denoising results in the supplementary file.

4.3.3 Experiments on Real Noisy Color Images

We evaluate the proposed method on two real noisy color image datasets, where the images were captured under indoor or outdoor lighting conditions by different types of cameras and camera settings. For WNNM-3 and MC-WNNM, the initial penalty parameter is set as $\rho_0 = 8$ and $\rho_0 = 6$, respectively. The number of iterations in Algorithm 2 is set as $K_2 = 2$.

Table 4.1: PSNR(dB) results of different denoising methods on the Kodak PhotoCD dataset.

	$\sigma_r = 40, \sigma_g = 20, \sigma_b = 30$									
Image#	**CBM3D**	**MLP**	**TNRD**	**DnCNN**	**NI**	**NC**	**WNNM-1**	**WNNM-2**	**WNNM-3**	**MC-WNNM**
1	25.24	25.70	25.74	20.47	23.85	24.90	26.01	25.95	25.58	**26.66**
2	28.27	30.12	30.21	20.47	25.90	25.87	30.08	30.11	29.80	**30.20**
3	28.81	31.19	31.49	20.53	26.00	28.58	31.58	31.61	31.20	**32.25**
4	27.95	29.88	29.86	20.47	25.82	25.67	30.13	30.16	29.84	**30.49**
5	25.03	26.00	26.18	20.52	24.38	25.15	26.44	26.39	25.32	**26.82**
6	26.24	26.84	26.90	20.66	24.65	24.74	27.39	27.30	26.88	**27.98**
7	27.88	30.28	30.40	20.52	25.63	27.69	30.47	30.54	29.70	**30.98**
8	25.05	25.59	25.83	20.57	24.02	25.30	26.71	26.75	25.26	**26.90**
9	28.44	30.75	30.81	20.50	25.94	27.44	30.86	30.92	30.29	**31.49**
10	28.27	30.38	30.57	20.52	25.87	28.42	30.65	30.68	29.95	**31.26**
11	26.95	28.00	28.14	20.52	25.32	24.67	28.19	28.16	27.61	**28.63**
12	28.76	30.87	31.05	20.60	26.01	28.37	30.97	31.06	30.58	**31.48**
13	23.76	23.95	23.99	20.52	23.53	22.76	24.27	24.15	23.52	**24.89**
14	26.02	26.97	27.11	20.51	24.94	25.68	27.20	27.15	26.55	**27.57**
15	28.38	30.15	30.44	20.71	26.06	28.21	30.52	30.60	30.13	**30.81**
16	27.75	28.82	28.87	20.52	25.69	26.66	29.27	29.21	29.02	**29.96**
17	27.90	29.57	29.80	20.56	25.85	28.32	29.78	29.79	29.16	**30.40**
18	25.77	26.40	26.41	20.53	24.74	25.70	26.63	26.56	26.01	**27.22**
19	27.30	28.67	28.81	20.53	25.40	26.52	29.19	29.22	28.67	**29.57**
20	28.96	30.40	30.76	21.44	24.95	25.90	30.79	30.83	29.97	**31.07**
21	26.54	27.53	27.60	20.51	25.06	26.48	27.80	27.75	27.12	**28.34**
22	27.05	28.17	28.27	20.51	25.36	26.60	28.21	28.16	27.81	**28.64**
23	29.14	32.31	32.51	20.54	26.13	23.24	31.89	31.97	31.21	**32.34**
24	25.75	26.41	26.53	20.59	24.55	25.73	27.10	27.03	26.18	**27.59**
Average	27.13	28.54	28.68	20.58	25.24	26.19	28.84	28.83	28.22	**29.31**

Table 4.2: PSNR(dB) results of different denoising methods on the Kodak PhotoCD dataset.

	$\sigma_r = 30, \sigma_g = 10, \sigma_b = 50$									
Image#	**CBM3D**	**MLP**	**TNRD**	**DnCNN**	**NI**	**NC**	**WNNM-1**	**WNNM-2**	**WNNM-3**	**MC-WNNM**
1	23.38	26.49	26.50	20.21	24.82	23.59	26.40	25.60	24.76	**27.81**
2	25.19	30.94	30.90	20.43	26.82	27.79	30.89	29.75	29.21	**30.96**
3	25.39	32.03	32.09	20.47	27.52	27.41	32.20	31.17	30.39	**32.89**
4	24.96	30.55	30.47	20.34	27.34	27.00	30.74	29.71	29.10	**31.19**
5	23.29	26.65	26.73	20.34	25.72	26.67	26.74	25.98	24.68	**27.60**
6	24.09	27.76	27.70	20.45	26.10	26.12	27.85	26.96	26.01	**29.15**
7	24.89	30.70	30.72	20.40	27.17	28.07	30.91	29.94	28.87	**31.37**
8	23.30	26.12	26.27	20.32	25.59	26.11	26.87	26.33	24.74	**27.44**
9	25.20	31.35	31.31	20.36	27.74	28.33	31.30	30.45	29.44	**32.08**
10	25.13	31.01	31.05	20.38	27.60	28.53	31.12	30.17	29.21	**31.83**
11	24.54	28.79	28.82	20.40	26.72	24.40	28.73	27.79	26.94	**29.60**
12	25.43	31.60	31.60	20.44	27.82	29.01	31.59	30.62	29.91	**32.11**
13	22.50	24.71	24.73	20.17	24.96	23.36	24.70	23.85	22.86	**25.96**
14	23.91	27.69	27.72	20.34	26.26	23.08	27.62	26.81	25.91	**28.57**
15	25.45	31.09	31.05	20.68	27.36	28.49	31.29	30.21	29.46	**31.39**
16	24.89	29.79	29.73	20.39	27.35	27.10	29.84	28.85	28.13	**31.10**
17	25.12	30.26	30.24	20.52	27.15	27.54	30.11	29.35	28.43	**31.08**
18	23.83	27.26	27.26	20.39	26.05	26.15	27.32	26.18	25.28	**28.32**
19	24.63	29.40	29.39	20.39	27.06	27.41	29.78	28.87	28.05	**30.53**
20	26.43	31.16	31.27	21.39	26.43	26.92	31.25	30.43	29.41	**31.55**
21	24.24	28.26	28.27	20.33	26.66	27.18	28.22	27.45	26.40	**29.29**
22	24.51	29.03	29.06	20.33	26.83	27.64	29.02	27.81	27.18	**29.57**
23	25.55	32.87	32.75	20.46	27.60	23.75	32.58	31.46	30.50	**32.34**
24	23.85	27.06	27.13	20.37	25.86	27.05	27.50	26.63	25.55	**28.32**
Average	24.57	29.27	29.28	20.43	26.69	26.61	29.36	28.43	27.52	**30.09**

Table 4.3: PSNR(dB) results of different denoising methods on the Kodak PhotoCD dataset.

	$\sigma_r = 5, \sigma_g = 30, \sigma_b = 15$									
Image#	**CBM3D**	**MLP**	**TNRD**	**DnCNN**	**NI**	**NC**	**WNNM-1**	**WNNM-2**	**WNNM-3**	**MC-WNNM**
1	27.25	28.06	28.62	24.99	25.00	29.55	28.16	27.95	28.15	**30.20**
2	29.70	31.30	32.70	25.09	27.80	29.69	32.54	31.60	31.73	**34.04**
3	30.34	31.98	34.07	25.37	28.02	31.93	33.91	33.68	33.52	**35.55**
4	29.47	31.10	32.56	25.14	27.70	32.56	32.68	31.85	31.90	**34.06**
5	27.31	28.59	29.35	25.18	26.14	30.00	28.83	29.00	28.91	**30.05**
6	28.20	29.10	29.90	25.27	26.15	28.81	29.55	29.46	29.62	**31.64**
7	29.73	31.60	33.46	25.40	27.22	31.63	33.09	33.29	32.86	**34.24**
8	27.47	28.16	28.91	25.12	25.34	30.16	29.15	29.24	29.03	**29.91**
9	30.07	31.63	33.55	25.33	27.86	31.54	33.19	33.20	32.95	**34.53**
10	29.96	31.37	33.20	25.33	27.74	33.44	32.98	33.02	32.74	**34.38**
11	28.73	29.85	30.87	25.23	26.98	30.16	30.45	30.14	30.21	**32.10**
12	30.20	31.50	33.31	25.40	27.97	31.69	33.22	32.71	32.65	**34.64**
13	26.18	26.69	26.98	24.81	25.14	27.97	26.49	26.42	26.62	**28.30**
14	27.86	29.07	29.87	25.16	26.67	29.21	29.36	29.14	29.30	**31.18**
15	29.91	31.58	33.13	25.47	28.04	31.17	33.22	32.34	32.36	**34.27**
16	29.29	30.35	31.54	25.26	27.46	32.18	31.34	31.05	31.21	**33.72**
17	29.50	31.09	32.52	25.37	27.81	32.80	32.09	32.00	31.85	**33.61**
18	27.72	28.74	29.36	25.10	26.57	28.63	28.88	28.76	28.89	**30.56**
19	28.98	30.18	31.35	25.24	27.25	29.79	31.34	30.77	30.95	**33.10**
20	30.63	31.78	33.27	26.08	27.89	29.52	33.00	32.55	32.58	**34.18**
21	28.50	29.58	30.54	25.18	26.86	30.99	30.02	30.03	30.03	**31.69**
22	28.61	29.78	30.82	25.14	27.19	30.50	30.47	29.82	30.10	**32.08**
23	30.60	32.66	35.06	25.33	28.17	32.82	34.72	34.37	33.94	**35.16**
24	27.97	28.81	29.61	25.12	26.01	30.75	29.47	29.35	29.39	**30.93**
Average	28.92	30.19	31.44	25.26	27.04	30.73	31.17	30.91	30.89	**32.67**

The first dataset is provided in [**ncwebsite**], which includes 20 real noisy images collected under uncontrolled outdoor environment. Since there is no “ground truth” of the noisy images, the objective measures such as PSNR cannot be computed on this dataset.

The second dataset is provided in [**crosschannel2016**], which includes noisy images of 11 static scenes. The noisy images were collected under controlled indoor environment. Each scene was shot 500 times under the same camera and camera setting. The mean image of the 500 shots is roughly taken as the “ground truth", with which the PSNR can be computed. Since the image size is very large (about 7000×5000) and the 11 scenes share repetitive contents, the authors of [**crosschannel2016**] cropped 15 smaller images of size 512×512 for experiments. Fig. **??** shows the contents of these images. Quantitative comparisons on the 15 cropped images will be reported.

(a)Noisy (b)CBM3D (c)MLP (d)TNRD (e)NI

(f)NC (g)WNNM-1 (h)WNNM-2 (i)WNNM-3 (j)MC-WNNM

Fig. 4.3: Denoised images of the real noisy image "Dog" [**ncwebsite**] by different methods. The estimated noise levels of R, G, and B channels are 16.8, 17.0, and 16.6, respectively. The images are better to be zoomed in on screen.

Results on Dataset [ncwebsite]

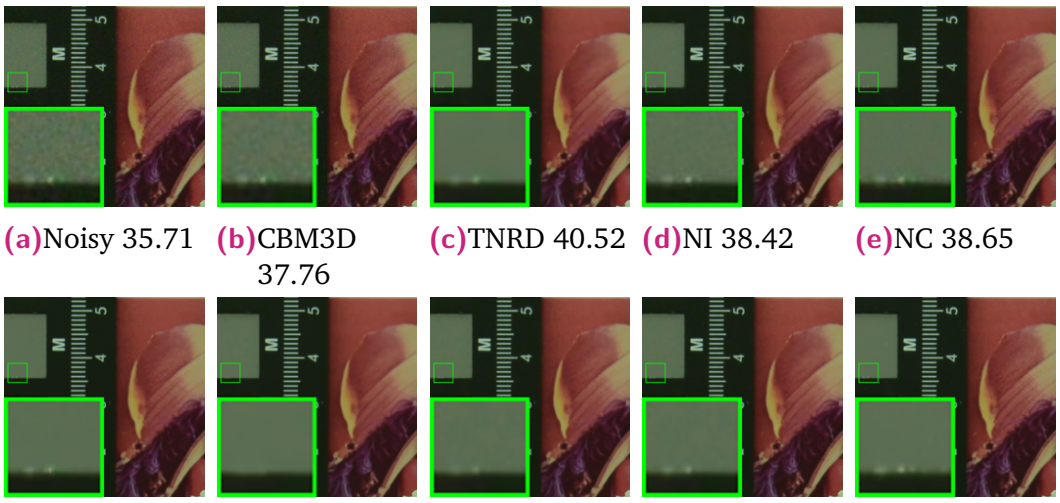
Since there is no "ground truth" for the real noisy images in dataset [**ncwebsite**], we only compare the visual quality of the denoised images by the compared methods. (Note that the method CC [**crosschannel2016**] is not compared here since its code is not publically available.)

Fig. **??** shows the denoised images of "Dog" by the competing methods. It can be seen that CBM3D, MLP, TRND and WNNM-1 tend to generate some noise caused color artifacts. Besides, WNNM-2 and WNNM-3 tend to over-smooth much the image. These results demonstrate that for color image denoising, neither processing each channel separately nor processing the three channels jointly but ignoring their noise difference is an effective solution. Though NC and NI methods are specifically developed for real color image denoising, their performance is not very satisfactory. In comparison, the proposed MC-WNNM recovers much better the structures and textures (such as the eye area) than the other competing methods. More visual comparisons on this dataset can be found in the supplementary file.

As described at the beginning of Section 4.3, there is a mean image for each noisy image in dataset [**crosschannel2016**], and those mean images can be roughly taken as "ground truth" for quantitative evaluation of denoising algorithms.

Table 4.4: PSNR(dB) results and averaged computational time (s) of different methods on 15 cropped real noisy images used in [**crosschannel2016**].

Camera Settings	CBM3D	MLP	TNRD	DnCNN	NI	NC	CC	WNNM-1	WNNM-2	WNNM-3	MC-W
	39.76	39.00	39.51	37.26	35.68	36.20	38.37	37.51	39.74	39.98	**41.**
Canon 5D	36.40	36.34	36.47	34.13	34.03	34.35	35.37	33.86	35.12	36.65	**37.**
ISO = 3200	36.37	36.33	36.45	34.09	32.63	33.10	34.91	31.43	33.14	34.63	**36.**
	34.18	34.70	34.79	33.62	31.78	32.28	34.98	33.46	35.08	35.08	**35.**
Nikon D600	35.07	36.20	36.37	34.48	35.16	35.34	35.95	36.09	36.42	36.84	**37.**
ISO = 3200	37.13	39.33	39.49	35.41	39.98	40.51	**41.15**	39.86	40.78	39.24	39.
	36.81	37.95	38.11	35.79	34.84	35.09	37.99	36.35	38.28	38.61	**39.**
Nikon D800	37.76	40.23	40.52	36.08	38.42	38.65	40.36	39.99	41.24	40.81	**41.**
ISO = 1600	37.51	37.94	38.17	35.48	35.79	35.85	38.30	37.15	38.04	38.96	**39.**
	35.05	37.55	37.69	34.08	38.36	38.56	39.01	38.60	**39.93**	37.97	38.
Nikon D800	34.07	35.91	35.90	33.70	35.53	35.76	36.75	36.04	37.32	37.30	**37.**
ISO = 3200	34.42	38.15	38.21	33.31	40.05	40.59	39.06	39.73	**41.52**	38.68	39.
	31.13	32.69	32.81	29.83	34.08	34.25	34.61	33.29	**35.20**	34.57	34.
Nikon D800	31.22	32.33	32.33	30.55	32.13	32.38	33.21	31.16	33.61	33.43	**33.**
ISO = 6400	30.97	32.29	32.29	30.09	31.52	31.76	33.22	31.98	33.62	**34.02**	33.
Average	35.19	36.46	36.61	33.86	35.33	35.65	36.88	35.77	37.27	37.12	**37.**
Time	7.8	20.4	6.7	180.3	**0.9**	18.2	NA	689.1	465.3	198.6	202



(a) Noisy 35.71 **(b)**CBM3D 37.76 **(c)**TNRD 40.52 **(d)**NI 38.42 **(e)**NC 38.65

(f)CC 40.36 **(g)**WNNM-2 41.24 **(h)**WNNM-3 40.81 **(i)**MC-WNNM **41.43** **(j)**Mean Image

Fig. 4.4: Denoised images and PSNR (dB) results of a region cropped from the real noisy image "Nikon D800 ISO=1600 2" [**crosschannel2016**] by different methods. The estimated noise levels of R, G, and B channels are 1.3, 1.1, and 1.4, respectively. The images are better to be zoomed in on screen.

The results on PSNR and averaged computational time by competing methods (including CC [**crosschannel2016**] whose results are copied from [**crosschannel2016**]) are listed in Table 4.4. For methods MLP [**mlp**] and TNRD [**chen2015learning**], both of them achieve the best results when setting the noise level of the trained models at $\sigma = 10$. The highest PSNR results are highlighted in bold. On average, MC-WNNM achieves 1.94dB, 0.44dB, 0.59dB improvements over the three WNNM methods, and significantly outperforms other competing method, including CC [**crosschannel2016**]. On 10 out of the 15 images, the proposed MC-WNNM achieves the highest PSNR values, while WNNM-2 achieves the highest PSNR results on 3 of 15 images. It should be noted that in the CC method [**crosschannel2016**], a specific model is trained for each camera and camera setting, while the other methods uses the same model for all cases.

Fig. **??** shows the denoised images of a scene captured by Nikon D800 ISO=1600. (The results of DnCNN and WNNM-1 are not shown here due to the limit of space.) We can see that CBM3D, NI, NC and CC will either remain noise or generate color artifacts, while TNRD, WNNM-2 and WNNM-3 over-smooth the image. In addition, due to treating each channel equally, both the denoised images (Fig. **??**(g) and Fig. **??**(h)) by WNNM-2 and WNNM-3 have chromatic aberration compared to the mean image (Fig. **??**(j)). MC-WNNM results in much better visual quality than other methods. More visual comparisons can be found in the supplementary file.

Comparison on speed. We compare the average computational time (second) of different methods (except CC), which is shown in Table 4.4. All experiments are run under the Matlab environment on a machine with 3.5GHz CPU and 32GB RAM. The fastest result is highlighted in bold. One can see that Neat Image (NI) is the fastest and costs about 0.9 second, while the proposed MC-WNNM needs 202.9 seconds. Noted that CBM3D, TNRD, and NC are implemented with compiled C++ mex-function and with parallelization, while WNNM, MLP, DnCNN, and the proposed MC-WNNM are implemented purely in Matlab.

4.4 Conclusion

The real noisy color images have different noise statistics across the R, G, B channels due to digital camera pipelines in CCD or CMOS sensors. This makes

the real color image denoising problem more challenging than grayscale image denoising. In this paper, we proposed a novel multi-channel (MC) denoising model to effectively exploit the redundancy across color channels while differentiating their different noise statistics. Specifically, we introduced a weight matrix to the data term in the RGB channel concatenated weighted nuclear norm minimization (WNNM) model, and the resulting MC-WNNM model can process adaptively the different noise in RGB channels. We solved the MC-WNNM model via an ADMM algorithm. Extensive experiments on synthetic and real datasets demonstrated that the proposed MC-WNNM method outperforms significantly the other competing denoising methods.

The proposed MC-WNNM model can be extended in at least two directions. Firstly, it is worthy to investigate new weight matrix beyond the diagonal form, such as the correlation form [**nearcor**], to further improve the color image denoising performance. Secondly, the proposed MC-WNNM model can be extended for hyperspectral image analysis, which may contain hundreds of bands with complex noise statistics.

5 Internal Nonlocal Self-Similarity Prior for Real Color Image Denoising: A Sparse Coding based ethod

Image denoising aims to recover the clean image $\boldsymbol{x}$ from its noisy observation $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n}$, where $\boldsymbol{n}$ is the corrupted noise. The denoising problem has been extensively studied in computer vision and machine learning, and numerous statistical image modeling and learning methods have been proposed in the past decades [**ksvd**; **bm3d**; **cbm3d**; **lssc**; **ncsr**; **wnnm**; **epll**; **pgpd**; **mlp**; **csf**; **chen2015learning**; **dncnn**; **Liu2008**; **noiseclinic**; **Zhu_2016_CVPR**; **crosschannel2016**; **neatimage**]. Most of the existing methods [**ksvd**; **bm3d**; **cbm3d**; **lssc**; **ncsr**; **wnnm**; **epll**; **pgpd**; **mlp**; **csf**; **chen2015learning**; **dncnn**] focus on additive white Gaussian noise (AWGN), which can be categorized into dictionary learning based methods [**ksvd**], nonlocal self-similarity based methods [**bm3d**; **cbm3d**; **lssc**; **ncsr**; **wnnm**], sparsity based methods [**ksvd**; **bm3d**; **cbm3d**; **lssc**; **ncsr**], low-rankness based methods [**wnnm**], generative learning based methods [**epll**; **pgpd**], and discriminative learning based methods [**mlp**; **csf**; **chen2015learning**; **dncnn**]. However, the realistic noise in real-world images is much more complex than AWGN [**crosschannel2016**] and varies with different cameras and camera settings (such as ISO, shutter speed, and aperture, etc.). Though have shown promising performance on AWGN noise removal, the above methods [**ksvd**; **bm3d**; **cbm3d**; **lssc**; **ncsr**; **wnnm**; **epll**; **pgpd**; **mlp**; **csf**; **chen2015learning**; **dncnn**] will become much less effective when dealing with complex realistic noise.

In the past decade, several denoising methods for realistic noisy images have been developed [**cbm3d**; **Liu2008**; **noiseclinic**; **Zhu_2016_CVPR**; **crosschannel2016**; **neatimage**]. Among them, CBM3D [**cbm3d**] is a representative method which applies the benchmark BM3D method [**bm3d**] to each channel of the luminance-chrominance space of the noisy image. Liu et al. [**Liu2008**] proposed to estimate the noise via a “noise level function” and

remove the noise for each channel of the real image. However, processing each channel separately would often achieve unsatisfactory performance and generate artifacts. The methods [**noiseclinic**; **Zhu_2016_CVPR**] perform image denoising by concatenating the patches of RGB channels into a vector. However, the concatenation does not consider the different noise statistics among different channels. The method in [**crosschannel2016**] models the noise in a noisy image as multivariate Gaussian and performs denoising by the Non-local Bayes filter [**noiseclinic**]. The commercial software Neat Image [**neatimage**] estimates the global noise parameters from a flat region of the given noisy image and filters the noise accordingly. However, both the two methods [**crosschannel2016**; **neatimage**] ignore the local statistical property of the noise which is signal dependent and varies in different pixels. By far, realistic image denoising remains a challenging problem in low level vision.

Sparse coding (SC) [**lasso**] has been well studied in many computer vision and learning problems [**wright2009robust**; **yang2009linear**; **yang2010image**], including image denoising [**ksvd**; **lssc**; **ncsr**]. Usually, the SC models employ an ℓ_2 (or Frobenious) norm to describe the residual in the data term, and minimize an energy function $\min_{\boldsymbol{\alpha}} \|\boldsymbol{y} - \boldsymbol{D}\boldsymbol{\alpha}\|_2^2 + \mathcal{R}(\boldsymbol{\alpha})$, where $\boldsymbol{D}$ is the dictionary, $\boldsymbol{\alpha}$ is the sparse coefficients vector of signal $\boldsymbol{y}$, and $\mathcal{R}(\boldsymbol{\alpha})$ is usually the ℓ_0 or ℓ_1 norm of $\boldsymbol{\alpha}$ to enforce sparse regularization. Some representative SC based image denoising methods include K-SVD [**ksvd**], LSSC [**lssc**], and NCSR [**ncsr**]. Though being effective on dealing with AWGN, SC based denoising methods are essentially limited by the ℓ_2 (or Frobenious) norm based data term, which actually assumes white Gaussian noise and is not able to characterize the complex realistic noise. Recently, there are several methods [**Zhu_2016_CVPR**; **zhao2014robust**] modeling complex noise via mixture of Gaussian distribution. However, these methods [**Zhu_2016_CVPR**; **zhao2014robust**] are time-consuming due to employing variational Bayesian inference techniques.

In this paper, we propose to lift the SC model to a robust denoiser for real noisy images by utilizing the channel-wise statistics and locally signal dependent property of the realistic noise. Specifically, we propose a trilateral weighted sparse coding (TWSC) scheme for realistic image denoising. Two weight matrices are introduced into the data term of the SC model to characterize the realistic noise property, and another weight matrix is introduced into the regularization term to characterize the sparsity priors of natural

images. To solve the proposed TWSC model, we reformulate it into a linear equality-constrained optimization program, which can be minimized under the alternating direction method of multipliers [**admm**] framework. Theoretical analysis on the existence and uniqueness of the solution of the TWSC model, as well as the convergence of the algorithm, are presented. Experiments on real noisy image datasets demonstrate that the proposed TWSC method achieves much more robust realistic denoising performance than the existing methods.

5.1 The Proposed Realistic Image Denoising Algorithm

5.1.1 The Trilateral Weighted Sparse Coding Model

Denote by $\boldsymbol{y}_c = \boldsymbol{x}_c + \boldsymbol{n}_c$ the noisy observation of the clean color image $\boldsymbol{x}_c$, where $c \in \{r, g, b\}$ is the index of R, G, B channels and $\boldsymbol{n}_c$ is the noise in channel c. A local patch of size $p \times p \times 3$ is extracted from the image and stretched to a vector, denoted by $\boldsymbol{y} = [\boldsymbol{y}_r^\top \ \boldsymbol{y}_g^\top \ \boldsymbol{y}_b^\top]^\top \in \mathbb{R}^{3p^2}$, where $\boldsymbol{y}_c \in \mathbb{R}^{p^2}$ is the corresponding patch in channel c. For each local patch $\boldsymbol{y}$, we search the M most similar patches to it (including $\boldsymbol{y}$ itself) by Euclidean distance in a local window around it. By stacking the M similar patches column by column, we form a noisy patch matrix $\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{N} \in \mathbb{R}^{3p^2 \times M}$, where $\boldsymbol{X}$ and $\boldsymbol{N}$ are the corresponding clean and noise patch matrices, respectively. The noisy patch matrix can be written as $\boldsymbol{Y} = [\boldsymbol{Y}_r^\top \ \boldsymbol{Y}_g^\top \ \boldsymbol{Y}_b^\top]^\top$, where $\boldsymbol{Y}_c$ is the sub-matrix of channel c.

Suppose that we have a dictionary $\boldsymbol{D} = [\boldsymbol{D}_r^\top \ \boldsymbol{D}_g^\top \ \boldsymbol{D}_b^\top]^\top$, where $\boldsymbol{D}_c$ is the sub-dictionary corresponding to channel c. Under the sparse coding (SC) framework [**lasso**], the sparse code matrix of $\boldsymbol{Y}$ over $\boldsymbol{D}$ can be obtained by

$$\hat{\boldsymbol{C}} = \arg\min_{\boldsymbol{C}} \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{C}\|_1, \tag{5.1}$$

where λ is the regularization parameter. Once $\hat{\boldsymbol{C}}$ is computed, the latent clean patch matrix $\hat{\boldsymbol{X}}$ can be estimated as $\hat{\boldsymbol{X}} = \boldsymbol{D}\hat{\boldsymbol{C}}$. In this paper, we use an adaptive dictionary to the given data matrix $\boldsymbol{Y}$. The dictionary $\boldsymbol{D}$ can be obtained by applying SVD to $\boldsymbol{Y}$ as

$$\boldsymbol{Y} = \boldsymbol{D}\boldsymbol{S}\boldsymbol{V}^\top. \tag{5.2}$$

Though SC based methods [**ksvd**; **lssc**; **ncsr**] have achieved promising performance in removing additive white Gaussian noise (AWGN), their performance is very limited when dealing with realistic noise in real-world images. The reason is that the realistic noise is non-Gaussian, varies locally and across channels, which cannot be characterized well by the Frobenious norm in model (5.1).

To account for the varying statistics of the realistic noise in different channels and different patches, we introduce two weight matrices $\boldsymbol{W}_1 \in \mathbb{R}^{3p^2\times 3p^2}$ and $\boldsymbol{W}_2 \in \mathbb{R}^{M\times M}$ to characterize the SC residual $(\boldsymbol{Y}-\boldsymbol{DC})$ in the data term of Eq. (5.1). Besides, to better characterize the sparse priors of the natural images, we introduce a third weight matrix $\boldsymbol{W}_3$, which is related to the distribution of the sparse coefficients matrix $\boldsymbol{C}$, into the regularization term of Eq. (5.1). Finally, the proposed trilateral weighted sparse coding (TWSC) model is formulated as follows:

$$\min_{\boldsymbol{C}} \|\boldsymbol{W}_1(\boldsymbol{Y}-\boldsymbol{DC})\boldsymbol{W}_2\|_F^2 + \|\boldsymbol{W}_3^{-1}\boldsymbol{C}\|_1. \tag{5.3}$$

All the three weight matrices are diagonal matrices and have clear physical meanings. $\boldsymbol{W}_1$ is a block diagonal matrix. Each block of $\boldsymbol{W}_1$ has the same diagonal elements to describe the noise properties in each RGB channel. The realistic noise in a local region could be approximately modeled as Gaussian [**Leungtip**], and each diagonal element of weight matrix $\boldsymbol{W}_2$ is used to describe the noise variance in each patch $\boldsymbol{y}$. Geometrically speaking, $\boldsymbol{W}_1$ is employed to regularize the row discrepancy of residual matrix $(\boldsymbol{Y}-\boldsymbol{DC})$, while $\boldsymbol{W}_2$ is employed to regularize the column discrepancy of $(\boldsymbol{Y}-\boldsymbol{DC})$. For matrix $\boldsymbol{W}_3$, each diagonal element will be set based on the prior information on $\boldsymbol{C}$. All the three weight matrices $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3$ can be determined under the *maximum a-posterior* (MAP) estimation framework, as described in detail in the next subsection.

5.1.2 The Setting of Weight Matrices

We determine the weight matrices $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3$ by employing the MAP estimation:

$$\hat{\boldsymbol{C}} = \arg\max_{\boldsymbol{C}} \ln P(\boldsymbol{C}|\boldsymbol{Y}) = \arg\max_{\boldsymbol{C}} \{\ln P(\boldsymbol{Y}|\boldsymbol{C}) + \ln P(\boldsymbol{C})\}. \tag{5.4}$$

The log-likelihood term $\ln P(\boldsymbol{Y}|\boldsymbol{C})$ is characterized by the statistics of noise or residual. According to [**Leungtip**], it can be assumed that the noise is independently and identically distributed (i.i.d.) in each channel and each patch with Gaussian distribution. We denote by σ_c the standard deviation (std) of noise in channel c, $c \in \{r, g, b\}$, and denote by σ_m the std of noise in a color patch m, $m \in \{1, ..., M\}$. Denote by σ_{cm} the noise std of sub-patch m in channel c. Then we have:

$$P(\boldsymbol{Y}|\boldsymbol{C}) = \prod_{c\in\{r,g,b\}} \prod_{m=1}^{M} (\pi\sigma_{cm})^{-p^2} \exp(-\sigma_{cm}^{-2}\|\boldsymbol{y}_{cm} - \boldsymbol{D}_c\boldsymbol{c}_m\|_2^2), \tag{5.5}$$

where $\boldsymbol{y}_{cm}, \boldsymbol{c}_m$ are the mth column of the matrices $\boldsymbol{Y}_c$ and $\boldsymbol{C}$, respectively. From the perspective of statistics [**glm**], the set of $\{\sigma_{cm}\}$ can be viewed as a $3 \times M$ contingency table created by two variables σ_c and σ_m, and their relationship could be modeled by a log-linear model $\sigma_{cm} = \sigma_c^{l_1}\sigma_m^{l_2}$, where $l_1 + l_2 = 1$. The estimation of $\{\sigma_{cm}\}$ can be reduced to the estimation of $\{\sigma_c\}$ and $\{\sigma_m\}$. Specifically, the noise std σ_c of channel c can be estimated by some noise estimation methods [**Chen2015ICCV**]. The noise std for the mth patch of $\boldsymbol{Y}$ can be initialized as $\sigma_m = \sigma \triangleq \sqrt{(\sigma_r^2 + \sigma_g^2 + \sigma_b^2)/3}$ and updated as $\sigma_m = \sqrt{\sigma^2 - \|\boldsymbol{y}_m - \boldsymbol{x}_m\|_2^2}$, where $\boldsymbol{y}_m$ is the mth image patch in $\boldsymbol{Y}$, and $\boldsymbol{x}_m = \boldsymbol{D}\boldsymbol{c}_m$ is the mth image patch recovered in previous iteration (refer to Algorithm 2 in next section).

We assume that each column $\boldsymbol{c}_m$ of coefficients matrix $\boldsymbol{C}$ follows i.i.d. Laplacian distribution. Specifically, for each entry $\boldsymbol{c}_m^i$, which is the coding coefficient of the mth patch over the ith atom of dictionary $\boldsymbol{D}$, we assume that it follows distribution $(2\boldsymbol{S}_i)^{-1}\exp(-\boldsymbol{S}_i^{-1}|\boldsymbol{c}_m^i|)$, where $\boldsymbol{S}_i$ is the ith element of the diagonal singular value matrix $\boldsymbol{S}$ in Eq. (5.2). Note that we set the scale factor of the distribution as the inverse of the ith singular value $\boldsymbol{S}_i$. This is because the larger the singular value $\boldsymbol{S}_i$ is, the more important the ith singular vector in $\boldsymbol{D}$ should be, and hence the distribution of the coding coefficients over this singular vector should have stronger regularization with less sparse distribution. The prior term in Eq. (5.4) becomes

$$P(\boldsymbol{C}) = \prod_{m=1}^{M} \prod_{i=1}^{3p^2} (2\boldsymbol{S}_i)^{-1} \exp(-\boldsymbol{S}_i^{-1}|\boldsymbol{c}_m^i|). \tag{5.6}$$

Put (5.6) and (5.5) into (5.4) and consider the log-linear model $\sigma_{cm} = \sigma_c^{l_1}\sigma_m^{l_2}$, we have

$$\begin{aligned}\hat{\boldsymbol{C}} &= \arg\min_{\boldsymbol{C}} \sum_{c\in\{r,g,b\}} \sum_{m=1}^{M} \sigma_{cm}^{-2} \|(\boldsymbol{y}_{cm} - \boldsymbol{D}_c\boldsymbol{c}_m)\|_2^2 + \sum_{m=1}^{M} \|\boldsymbol{S}^{-1}\boldsymbol{c}_m\|_1 \\ &= \arg\min_{\boldsymbol{C}} \sum_{c\in\{r,g,b\}} \sigma_c^{-2l_1} \|(\boldsymbol{Y}_c - \boldsymbol{D}_c\boldsymbol{C})\boldsymbol{W}_2\|_F^2 + \|\boldsymbol{S}^{-1}\boldsymbol{C}\|_1 \\ &= \arg\min_{\boldsymbol{C}} \|\boldsymbol{W}_1(\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{C})\boldsymbol{W}_2\|_F^2 + \|\boldsymbol{W}_3^{-1}\boldsymbol{C}\|_1,\end{aligned} \tag{5.7}$$

where

$$\boldsymbol{W}_1 = \text{diag}(\sigma_r^{-l_1}\boldsymbol{I}_{p^2}, \sigma_g^{-l_1}\boldsymbol{I}_{p^2}, \sigma_b^{-l_1}\boldsymbol{I}_{p^2}), \boldsymbol{W}_2 = \text{diag}(\sigma_1^{-l_2}, ..., \sigma_M^{-l_2}), \boldsymbol{W}_3 = \boldsymbol{S}, \tag{5.8}$$

and $\boldsymbol{I}_{p^2}$ is the p^2 dimensional identity matrix. We can see that the diagonal elements of $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are determined by the noise standard deviation in each channel and each patch, respectively. The stronger the noise in a channel and a patch, the less the contribution that channel and patch will make to the denoising output. In our experiments, we consider $\{\sigma_c\}, \{\sigma_m\}$ of equal importance and empirically set $l_1 = l_2 = 0.5$.

5.1.3 Model Optimization

Letting $\boldsymbol{C}^* = \boldsymbol{W}_3^{-1}\boldsymbol{C}$, we can transfer the weight matrix $\boldsymbol{W}_3$ into the data term of (5.3). Thus, the TWSC model (5.3) is reformulated as

$$\min_{\boldsymbol{C}^*} \|\boldsymbol{W}_1(\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{W}_3\boldsymbol{C}^*)\boldsymbol{W}_2\|_F^2 + \|\boldsymbol{C}^*\|_1. \tag{5.9}$$

To make the notation simple, we remove the superscript $*$ in $\boldsymbol{C}^*$ and still use $\boldsymbol{C}$ in the following development.

Since the problem (5.9) is convex, we can obtain its globally optimal solution. We employ the variable splitting method [**Eckstein1992**] to solve it. By introducing an augmented variable $\boldsymbol{Z}$, the problem (5.9) is reformulated as a linear equality-constrained problem with two variables $\boldsymbol{C}$ and $\boldsymbol{Z}$:

$$\min_{\boldsymbol{C},\boldsymbol{Z}} \|\boldsymbol{W}_1(\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{W}_3\boldsymbol{C})\boldsymbol{W}_2\|_F^2 + \|\boldsymbol{Z}\|_1 \quad \text{s.t.} \quad \boldsymbol{C} = \boldsymbol{Z}. \tag{5.10}$$

Since the objective function is separable w.r.t. the two variables, the problem (5.10) can be solved under the alternating direction method of multipliers (ADMM) [**admm**] framework. The augmented Lagrangian function is:

$$\mathcal{L}(\boldsymbol{C}, \boldsymbol{Z}, \boldsymbol{\Delta}, \rho) = \|\boldsymbol{W}_1(\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{W}_3\boldsymbol{C})\boldsymbol{W}_2\|_F^2 + \|\boldsymbol{Z}\|_1 + \langle \boldsymbol{\Delta}, \boldsymbol{C} - \boldsymbol{Z}\rangle + \frac{\rho}{2}\|\boldsymbol{C} - \boldsymbol{Z}\|_F^2, \tag{5.11}$$

where $\boldsymbol{\Delta}$ is the augmented Lagrangian multiplier and $\rho > 0$ is the penalty parameter. We initialize the matrix variables $\boldsymbol{C}_0$, $\boldsymbol{Z}_0$, and $\boldsymbol{\Delta}_0$ to be comfortable zero matrix and $\rho_0 > 0$. Denote by $(\boldsymbol{C}_k, \boldsymbol{Z}_k)$ and $\boldsymbol{\Delta}_k$ the optimization variables and Lagrange multiplier at iteration k ($k = 0, 1, 2, ...$), respectively. By taking derivatives of the Lagrangian function $\mathcal{L}$ w.r.t. $\boldsymbol{C}$ and $\boldsymbol{Z}$ and setting the derivatives to be zeros, we can alternatively update the variables as follows:

(1) **Update $\boldsymbol{C}$ by fixing $\boldsymbol{Z}$ and $\boldsymbol{\Delta}$**:

$$\boldsymbol{C}_{k+1} = \arg\min_{\boldsymbol{C}} \|\boldsymbol{W}_1(\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{W}_3\boldsymbol{C})\boldsymbol{W}_2\|_F^2 + \frac{\rho_k}{2}\|\boldsymbol{C} - \boldsymbol{Z}_k + \rho_k^{-1}\boldsymbol{\Delta}_k\|_F^2. \tag{5.12}$$

This is a two-sided weighted least squares regression problem with the solution satisfying that

$$\boldsymbol{A}\boldsymbol{C}_{k+1} + \boldsymbol{C}_{k+1}\boldsymbol{B}_k = \boldsymbol{E}_k, \tag{5.13}$$

where $\boldsymbol{A} = \boldsymbol{W}_3^\top \boldsymbol{D}^\top \boldsymbol{W}_1^\top \boldsymbol{W}_1 \boldsymbol{D}\boldsymbol{W}_3$, $\boldsymbol{B}_k = \frac{\rho_k}{2}(\boldsymbol{W}_2\boldsymbol{W}_2^\top)^{-1}$, and $\boldsymbol{E}_k = \boldsymbol{W}_3^\top \boldsymbol{D}^\top \boldsymbol{W}_1^\top \boldsymbol{W}_1 \boldsymbol{Y} + (\frac{\rho_k}{2}\boldsymbol{Z}_k - \frac{1}{2}\boldsymbol{\Delta}_k)(\boldsymbol{W}_2\boldsymbol{W}_2^\top)^{-1}$. Eq. (8.60) is a standard Sylvester equation (SE) which has a unique solution if and only if $\sigma(\boldsymbol{A}) \cap \sigma(-\boldsymbol{B}_k) = \emptyset$, where $\sigma(\boldsymbol{F})$ denotes the spectrum, i.e., the set of eigenvalues, of the matrix $\boldsymbol{F}$ [**simoncini2016computational**]. We can rewrite the SE (8.60) as

$$(\boldsymbol{I}_M \otimes \boldsymbol{A} + \boldsymbol{B}_k^\top \otimes \boldsymbol{I}_{3p^2})\text{vec}(\boldsymbol{C}_{k+1}) = \text{vec}(\boldsymbol{E}_k), \tag{5.14}$$

and the solution $\boldsymbol{C}_{k+1}$ (if existed) can be obtained via $\boldsymbol{C}_{k+1} = \text{vec}^{-1}(\text{vec}(\boldsymbol{C}_{k+1}))$, where $\text{vec}^{-1}(\bullet)$ is the inverse of the vec-operator $\text{vec}(\bullet)$. Detailed theoretical analysis on the existence of the unique solution is given in Section 3.

(2) **Update $\boldsymbol{Z}$ by fixing $\boldsymbol{C}$ and $\boldsymbol{\Delta}$**:

$$\boldsymbol{Z}_{k+1} = \arg\min_{\boldsymbol{Z}} \frac{\rho_k}{2}\|\boldsymbol{Z} - (\boldsymbol{C}_{k+1} + \rho_k^{-1}\boldsymbol{\Delta}_k)\|_F^2 + \|\boldsymbol{Z}\|_1. \tag{5.15}$$

This problem has a closed-form solution as

$$\boldsymbol{Z}_{k+1} = \mathcal{S}_{\rho_k^{-1}}(\boldsymbol{C}_{k+1} + \rho_k^{-1}\boldsymbol{\Delta}_k), \tag{5.16}$$

where $\mathcal{S}_\lambda(x) = \text{sign}(x)\max(x-\lambda, 0)$ is the soft-thresholding operator.

(3) **Update $\boldsymbol{\Delta}$ by fixing $\boldsymbol{X}$ and $\boldsymbol{Z}$**:

$$\boldsymbol{\Delta}_{k+1} = \boldsymbol{\Delta}_k + \rho_k(\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}). \tag{5.17}$$

(4) **Update ρ**: $\rho_{k+1} = \mu * \rho_k$, where $\mu > 1$.

The above alternative updating steps are repeated until the convergence condition is satisfied or the number of iterations exceeds a preset threshold K_1. The ADMM algorithm converges when $\|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F \leq$ Tol, $\|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F \leq$ Tol, and $\|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F \leq$ Tol are simultaneously satisfied, where Tol > 0 is a small tolerance number. We summarize the updating procedures in Algorithm 1. The convergence analysis of Algorithm 1 is given in Theorem 7. Note that here we employ an unbounded sequence of $\{\rho_k\}$, i.e., $\lim_{k\to\infty} \rho_k = +\infty$, to make sure that Algorithm 1 converges.

5.1.4 Convergence Analysis

Theorem 4. *Given $\rho_{k+1} = \mu\rho_k$ ($\rho_0 > 0$) for $k \geq 0$ and $\mu > 1$, the sequences $\{\boldsymbol{C}_k\}$ and $\{\boldsymbol{Z}_k\}$ generated in Algorithm 1 satisfy:*

$$(a)\ \|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F = \mathcal{O}(\rho_k^{-1})\ \textit{as}\ k \to +\infty,\ \textit{i.e.},\ \lim_{k\to+\infty} \|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F = 0; \tag{5.18}$$

$$(b)\ \textit{If}\ \|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F = \mathcal{O}(\rho_k^{-1})\ \textit{as}\ k \to +\infty,\ \textit{then}\ \lim_{k\to+\infty} \|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F = 0; \tag{5.19}$$

$$(c)\ \textit{If}\ \lim_{k\to+\infty} \|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F = 0,\ \textit{then}\ \lim_{k\to+\infty} \|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F = 0. \tag{5.20}$$

The proof of Theorem 7 can be found in the supplementary file. Though Theorem 7 could not directly guarantee $\lim_{k\to+\infty} \|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F = 0$ and $\lim_{k\to+\infty} \|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F = 0$, it is empirically found that both $\|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F$ and $\|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F$ will approach to 0 simultaneously in all our tests. In Figure 5.1, we can see that the maximal errors in $|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}|, |\boldsymbol{C}_{k+1} - \boldsymbol{C}_k|, |\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k|$ approach to 0 simultaneously in 50 iterations.

Algorithm 1: Solve Eq. (5.10) via ADMM

Input: $\boldsymbol{Y}, \boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3$, μ, Tol, K_1;
Initialization: $\boldsymbol{C}_0 = \boldsymbol{Z}_0 = \boldsymbol{\Delta}_0 = \boldsymbol{0}$,
$\rho_0 > 0$, T = False, $k = 0$;
While (T == false) **do**
1. Update $\boldsymbol{C}_{k+1}$ by solving Eq. (8.60);
2. Update $\boldsymbol{Z}_{k+1}$ by soft thresholding (5.16);
3. Update $\boldsymbol{\Delta}_{k+1}$ as
$\boldsymbol{\Delta}_{k+1} = \boldsymbol{\Delta}_k + \rho_k(\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1})$;
4. Update $\rho_{k+1} = \mu * \rho_k$;
5. $k \leftarrow k + 1$;
if (Converged) or ($k \geq K_1$)
6. T $\leftarrow$ True
end if
end while
Output: Matrices $\boldsymbol{C}$ and $\boldsymbol{Z}$.

Algorithm 2: Image Denoising by TWSC

Input: Noisy image $\boldsymbol{y}_c$, $\{\sigma_r, \sigma_g, \sigma_b\}$, K_2;
Initialization: $\hat{\boldsymbol{x}}_c^{(0)} = \boldsymbol{y}_c$, $\boldsymbol{y}_c^{(0)} = \boldsymbol{y}_c$;
for $k = 1 : K_2$ **do**
1. Set $\boldsymbol{y}_c^{(k)} = \hat{\boldsymbol{x}}_c^{(k-1)}$;
2. Extract local patches $\{\boldsymbol{y}_j\}_{j=1}^N$ from $\boldsymbol{y}_c^{(k)}$;
for each patch $\boldsymbol{y}_j$ **do**
3. Search nonlocal similar patches $\boldsymbol{Y}_j$;
4. Apply the TWSC model (5.3) to $\boldsymbol{Y}_j$ and obtain the estimated $\boldsymbol{X}_j = \boldsymbol{DC}$;
end for
5. Aggregate $\{\boldsymbol{X}_j\}_{j=1}^N$ to form the image $\hat{\boldsymbol{x}}_c^{(k)}$;
end for
Output: Denoised image $\hat{\boldsymbol{x}}_c^{(K_2)}$.

5.1.5 The Denoising Algorithm

Given a noisy image $\boldsymbol{y}_c$, suppose that we have extracted N local patches $\{\boldsymbol{y}_j\}_{j=1}^N$ and their similar patches. Then N noisy patch matrices $\{\boldsymbol{Y}_j\}_{j=1}^N$ can be formed to estimate the clean matrices $\{\boldsymbol{X}_j\}_{j=1}^N$. The patches in matrices $\{\boldsymbol{X}_j\}_{j=1}^N$ are aggregated to form the denoised image $\hat{\boldsymbol{x}}_c$. To obtain better denoising results, we perform the above denoising procedures for several iterations. The proposed TWSC based robust image denoising algorithm is summarized in Algorithm 2.

5.2 Existence and Faster Solution of Sylvester Equation (8.60)

The solution of the Sylvester equation (SE) (8.60) does not always exist, though the solution is unique if it exists. Besides, solving SE (8.60) is usually computationally expensive in high dimensional cases. In this section, we provide a sufficient condition to guarantee the existence of the solution to SE (8.60), as well as a faster solution of (8.60) to save the computational cost of Algorithms 1 and 2.

5.2.1 Existence of the Unique Solution

Before we prove the existence of unique solution of Eq. (8.60), we first introduce the following theorem.

Theorem 5. *Assume that* $\boldsymbol{A} \in \mathbb{R}^{3p^2 \times 3p^2}$, $\boldsymbol{B} \in \mathbb{R}^{M \times M}$ *are both symmetric and positive semi-definite matrices. If at least one of* $\boldsymbol{A}, \boldsymbol{B}$ *is positive definite, the Sylvester equation* $\boldsymbol{AC} + \boldsymbol{CB} = \boldsymbol{E}$ *has a unique solution for* $\boldsymbol{C} \in \mathbb{R}^{3p^2 \times M}$.

The proof of Theorem 8 can be found in the supplementary file. We then have the following corollary.

Corollary 1. *The Sylvester equation (8.60) has a unique solution.*

Proof. Since $\boldsymbol{A}, \boldsymbol{B}_k$ in (8.60) are both symmetric and positive definite matrices, according to Theorem 8, the SE (8.60) has a unique solution. □

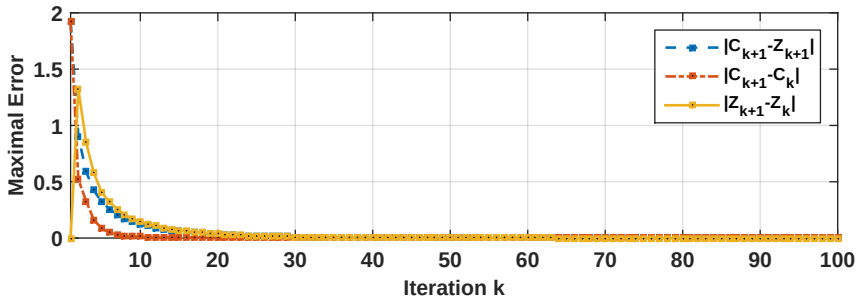


Fig. 5.1: The convergence curves of maximal errors in entries of $|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}|$ (blue line), $|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k|$ (red line), and $|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k|$ (yellow line). The test image is "Barbara".

5.2.2 Faster Solution of the Sylvester Equation (8.60)

The solution (5.14) of the SE (8.60) is typically obtained by the Bartels-Stewart algorithm [**Bartels1972**]. This algorithm firstly employs a QR factorization [**GolubMatrix**], implemented via Gram-Schmidt process, to decompose the matrices $\boldsymbol{A}$ and $\boldsymbol{B}_k$ into Schur forms, and then solves the obtained triangular system by the back-substitution method [**bareiss1968sylvester**]. However, since the matrices $\boldsymbol{I}_M \otimes \boldsymbol{A}$ and $\boldsymbol{B}_k^\top \otimes \boldsymbol{I}_{3p^2}$ are of $3p^2M \times 3p^2M$ dimensions, it is very computationally expensive ($\mathcal{O}(p^6M^3)$) to calculate their QR factorization to obtain the Schur forms. By exploiting the specific properties of our problem, we provide a faster while exact solution for the SE (8.60).

Since the matrices $\boldsymbol{A}, \boldsymbol{B}_k$ in (8.60) are symmetric and positive definite, the matrix $\boldsymbol{A}$ can be eigen-decomposed as $\boldsymbol{A} = \boldsymbol{U}_{\boldsymbol{A}}\boldsymbol{\Sigma}_{\boldsymbol{A}}\boldsymbol{U}_{\boldsymbol{A}}^\top$, with computational cost of $\mathcal{O}(p^6)$. Left multiply both sides of the SE (8.60) by $\boldsymbol{U}_{\boldsymbol{A}}^\top$, we can get $\boldsymbol{\Sigma}_A\boldsymbol{U}_{\boldsymbol{A}}^\top\boldsymbol{C}_{k+1} + \boldsymbol{U}_{\boldsymbol{A}}^\top\boldsymbol{C}_{k+1}\boldsymbol{B}_k = \boldsymbol{U}_{\boldsymbol{A}}^\top\boldsymbol{E}_k$. This can be viewed as an SE w.r.t. the matrix $\boldsymbol{U}_{\boldsymbol{A}}^\top\boldsymbol{C}_{k+1}$, with a unique solution $\text{vec}(\boldsymbol{U}_{\boldsymbol{A}}^\top\boldsymbol{C}_{k+1}) = (\boldsymbol{I}_M \otimes \boldsymbol{\Sigma}_{\boldsymbol{A}} + \boldsymbol{B}_k^\top \otimes \boldsymbol{I}_{3p^2})^{-1}\text{vec}(\boldsymbol{U}_{\boldsymbol{A}}^\top\boldsymbol{E}_k)$. Since the matrix $(\boldsymbol{I}_M \otimes \boldsymbol{\Sigma}_{\boldsymbol{A}} + \boldsymbol{B}_k^\top \otimes \boldsymbol{I}_{3p^2})$ is diagonal and positive definite, its inverse can be calculated on each diagonal element of $(\boldsymbol{I}_M \otimes \boldsymbol{\Sigma}_{\boldsymbol{A}} + \boldsymbol{B}_k^\top \otimes \boldsymbol{I}_{3p^2})$. The computational cost for this step is $\mathcal{O}(p^2M)$. Finally, the solution $\boldsymbol{C}_{k+1}$ can be obtained via $\boldsymbol{C}_{k+1} = \boldsymbol{U}_{\boldsymbol{A}}\text{vec}^{-1}(\text{vec}(\boldsymbol{U}_{\boldsymbol{A}}^\top\boldsymbol{C}_{k+1}))$. By this way, the complexity for solving the SE (8.60) is reduced from $\mathcal{O}(p^6M^3)$ to $\mathcal{O}(\max(p^6, p^2M))$, which is a huge computational saving.

5.3 Experiments

To validate the effectiveness of our proposed TWSC scheme, we apply it to both synthetic AWGN corrupted images and realistic noisy images. To better demonstrate the roles of the weights in our model, we compare with a baseline method, in which the weights $\boldsymbol{W}_1, \boldsymbol{W}_2$ are set to comfortable identity matrices, while the matrix $\boldsymbol{W}_3$ is set as in (5.8). We call this baseline the weighted sparse coding (WSC) method.

Implementation Details. We empirically set the parameter $\rho_0 = 0.5$ and $\mu = 1.1$. The maximum number of iteration is set as $K_1 = 100$. The window size for similar patch searching is set as 60×60. For parameters p, M, K_2, we set $p = 7, M = 70, K_2 = 8$ for $0 < \sigma \leq 20$; $p = 8, M = 90, K_2 = 12$ for $20 <$

$\sigma \leq 40$; $p = 8, M = 120, K_2 = 12$ for $40 < \sigma \leq 60$; $p = 9, M = 140, K_2 = 14$ for $60 < \sigma \leq 100$. All parameters are fixed in our experiments, which are run under the Matlab2014b environment on a machine with Intel(R) Core(TM) i7-5930K CPU of 3.5GHz and 32GB RAM. It takes about 240 seconds to process a real noisy image of size $512 \times 512 \times 3$.

5.3.1 Results on Additive White Gaussian Noise Removal

We first compare the proposed TWSC with the state-of-the-art AWGN denoising methods such as BM3D [**bm3d**], LSSC [**lssc**], NCSR [**ncsr**], WNNM [**wnnm**], TNRD [**chen2015learning**], and DnCNN [**dncnn**] on 20 gray level images commonly used in [**bm3d**]. Since TNRD and DnCNN are discriminative learning based methods, we retrain the two methods for noise standard deviations $5 \sim 100$ with a gap of 5 by using the source codes provided by the authors. The noisy images are generated by adding AWGN to each image with $\sigma = 20, 40, 60, 80, 100$, respectively. Note that in this experiment the weight matrix $\boldsymbol{W}_1 = \boldsymbol{I}_{p^2}$ is an identity matrix because the input images are gray level.

The averaged PSNR and SSIM [**ssim**] results are listed in Table 5.1. One can see that the proposed TWSC is only a little inferior to TNRD and DnCNN when $\sigma \leq 40$. Note that TNRD and DnCNN are trained on external and synthetic clean and noisy image pairs, which is unfair for comparison since TWSC only utilizes the tested noisy image. Besides, one can see that the proposed TWSC model works much better than the baseline method WSC, which proves that the weight matrix $\boldsymbol{W}_2$ can characterize better the noise statistics in local image patches. Due to limited space, we leave the visual comparisons of different methods in the supplementary file.

5.3.2 Results on Realistic Noise Removal

We evaluate the proposed method on two real noisy image datasets, where the images were captured under indoor and outdoor lighting conditions by different types of cameras and camera settings.

Dataset 1 is provided in [**ncwebsite**], which includes 20 real noisy images collected under uncontrolled environment. Since there is no "ground truth"

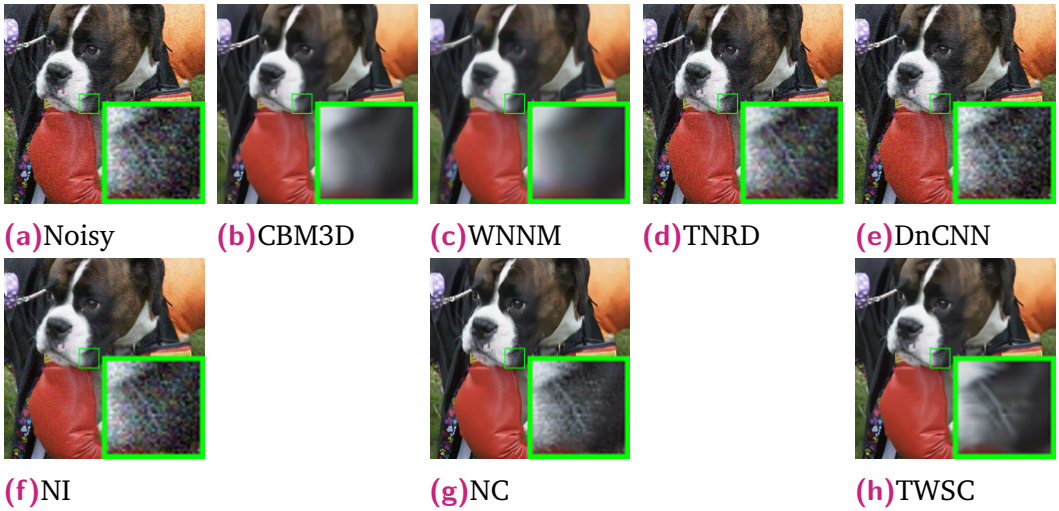
of the noisy images, we only compare the visual quality of the denoised images by different methods.

Dataset 2 is provided in [**crosschannel2016**], which includes noisy images of 11 static scenes. The noisy images were collected under controlled indoor environment. Each scene was shot 500 times under the same camera and camera setting. The mean image of the 500 shots is roughly taken as the "ground truth", with which the PSNR and SSIM [**ssim**] can be computed. 15 images of size 512×512 were cropped in [**crosschannel2016**] to evaluate the different denoising methods.

We compare the proposed TWSC method with CBM3D [**cbm3d**], WNNM [**wnnm**], TNRD [**chen2015learning**], DnCNN [**dncnn**], the commercial software Neat Image (NI) [**neatimage**], the state-of-the-art real image denoising methods "Noise Clinic" (NC) [**noiseclinic**] and CC [**crosschannel2016**]. Since WNNM and TNRD are designed for gray level images, we applied them to each channel of real noisy images. The input noise stds σ_c ($c \in \{r, g, b\}$) for WNNM are estimated by a noise estimation method [**Chen2015ICCV**]. TNRD achieves its best results when setting the noise std of the trained models at $\sigma_c = 10$. The methods of CBM3D and DnCNN can directly deal with color images, and the input noise std is set as $\sigma = \sqrt{(\sigma_r^2 + \sigma_g^2 + \sigma_b^2)/3}$. Due to limited space, we do not compare with the baseline method WSC on visual quality.

Results on Dataset 1. Fig. 4.2 show the denoised images of "Dog". (The method CC [**crosschannel2016**] is not compared since its code is not available.) One can see that CBM3D and WNNM tend to over-smooth the image. DnCNN, TNRD and NI remain many noise-caused color artifacts across the whole image. NC is better than other methods except the proposed TWSC. These results demonstrate that the methods designed for AWGN are not effective for realistic noise removal. Though NC and NI methods are specifically developed for real noisy images, their performance is not satisfactory. In comparison, the proposed TWSC works much better in removing the noise while maintaining the details (see the zoom-in window in "Dog") than the other competing methods. More visual comparisons can be found in the supplementary file.

Results on Dataset 2. The average PSNR and SSIM results on the 15 cropped images by competing methods are listed in Table 5.2. One can see that the proposed TWSC is much better than other competing methods, including CC



(a)Noisy (b)CBM3D (c)WNNM (d)TNRD (e)DnCNN

(f)NI (g)NC (h)TWSC

Fig. 5.2: Denoised images of the real noisy image *Dog* [**ncwebsite**] by different methods. The estimated noise levels of R, G, and B channels are 16.8, 17.0, and 16.6, respectively. The images are better to be zoomed in on screen.

and the baseline method WSC. Fig. 5.3 shows the denoised images of a scene captured by Canon 5D Mark 3 at ISO = 3200. One can see that the proposed TWSC method results in not only higher PSNR and SSIM measures, but also much better visual quality than the other denoising methods.

5.4 Conclusion

The realistic noise in real-world noisy images is very complex due to the various factors in digital camera pipelines, making the realistic image denoising problem much more challenging than white Gaussian noise removal. We proposed a novel trilateral weighted sparse coding (TWSC) scheme to exploit the noise properties across different channels and patches. Specifically, we introduced two weight matrices into the data term of sparse coding model to adaptively process each patch in each channel, and a weight matrix to better model image priors. The proposed TWSC model was solved via the ADMM algorithm and the solution existence and convergence can be guaranteed. Experiments demonstrated the superior performance of TWSC to the state-of-the-art denoising methods, including those methods designed for realistic noise.

Table 5.1: Average results on PSNR(dB) and SSIM of different denoising algorithms on 20 gray level images corrupted by AWGN noise.

σ_n	Metric	**BM3D**	**LSSC**	**NCSR**	**WNNM**	**TNRD**	**DnCNN**	**WSC**	**TWSC**
20	PSNR	30.95	30.95	30.95	31.20	31.30	31.36	30.90	31.15
	SSIM	0.8552	0.8561	0.8536	0.8580	0.8602	0.8631	0.8484	0.8581
40	PSNR	27.69	27.80	27.71	28.03	28.09	28.19	27.71	28.06
	SSIM	0.7726	0.7750	0.7717	0.7784	0.7824	0.7877	0.7682	0.7833
60	PSNR	26.02	25.83	25.82	26.25	26.14	26.35	25.93	26.28
	SSIM	0.7176	0.7086	0.7163	0.7249	0.7203	0.7307	0.7142	0.7309
80	PSNR	24.76	24.55	24.50	25.01	24.44	24.60	24.67	25.03
	SSIM	0.6716	0.6624	0.6738	0.6838	0.6528	0.6516	0.6727	0.6891
100	PSNR	23.78	23.59	23.49	24.03	22.56	17.07	23.62	24.06
	SSIM	0.6336	0.6299	0.6388	0.6455	0.4766	0.2367	0.6371	0.6538

Table 5.2: Average results on PSNR(dB) and SSIM of different denoising methods on 15 cropped real noisy images used in [**crosschannel2016**].

Metric	**CBM3D**	**WNNM**	**TNRD**	**DnCNN**	**NI**	**NC**	**CC**	**WSC**	**TWSC**
PSNR	35.19	35.77	36.61	33.86	35.49	36.43	36.88	37.36	**37.81**
SSIM	0.9063	0.9381	0.9463	0.8635	0.9126	0.9364	0.9481	0.9516	**0.9586**

(a) Mean Image **(b)**Noisy 37.00 **(c)**CBM3D 39.72 **(d)**WNNM 37.48 **(e)**TNRD 39.46

(f) DnCNN 37.26 **(g)**NI 37.68 **(h)**NC 38.76 **(i)**CC 38.37 **(j)**TWSC **40.75**

Fig. 5.3: Denoised images and PSNR (dB) results of the real noisy image *Canon 5D Mark 3 ISO 3200 1* [**crosschannel2016**] by different methods. The images are better to be zoomed in on screen.

6 A Large Real Noisy Image Dataset, with A Comprehensive Evaluation of State-of-the-arts

> *Users do not care about what is inside the box, as long as the box does what they need done.*
>
> — **Jef Raskin**
> about Human Computer Interfaces

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

6.1 Introduction

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

6.2 Related Work

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

6.3 Summary

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

7 Conclusions

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

7.1 Section 1

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

7.2 Section 2

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language.

There is no need for special content, but the length of words should match the language.

7.3 Future Work

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

8 Appendix

8.1 Closed-Form Solution of the Weighted Sparse Coding Problem (3.7)

For notation simplicity, we ignore the indices n, m, t in problem (3.7). It turns into the following weighted sparse coding problem:

$$\min_{\boldsymbol{\alpha}} \|\mathbf{y} - \boldsymbol{D}\boldsymbol{\alpha}\|_2^2 + \sum_{j=1}^{3p^2} \lambda_j |\boldsymbol{\alpha}_j|. \tag{8.1}$$

Since $\boldsymbol{D}$ is an orthogonal matrix, problem (8.1) is equivalent to:

$$\min_{\boldsymbol{\alpha}} \|\boldsymbol{D}^T\mathbf{y} - \boldsymbol{\alpha}\|_2^2 + \sum_{j=1}^{3p^2} \lambda_j |\boldsymbol{\alpha}_j|. \tag{8.2}$$

For simplicity, we denote $\mathbf{z} = \mathbf{D^T y}$. Here we have $\lambda_j > 0$, $j = 1, ..., 3p^2$, then problem (8.2) can be written as:

$$\min_{\boldsymbol{\alpha}} \sum_{j=1}^{3p^2} ((\mathbf{z}_j - \boldsymbol{\alpha}_j)^2 + \lambda_j |\boldsymbol{\alpha}_j|). \tag{8.3}$$

The problem (8.3) is separable w.r.t. each $\boldsymbol{\alpha}_j$ and hence can be simplified to $3p^2$ independent scalar minimization problems:

$$\min_{\boldsymbol{\alpha}_j} (\mathbf{z}_j - \boldsymbol{\alpha}_j)^2 + \lambda_j |\boldsymbol{\alpha}_j|, \tag{8.4}$$

where $j = 1, ..., 3p^2$. Taking derivative of $\boldsymbol{\alpha}_j$ in problem (8.4) and setting the derivative to be zero. There are two cases for the solution.

(a) If $\boldsymbol{\alpha}_j \geq 0$, we have $2(\boldsymbol{\alpha}_j - \mathbf{z}_j) + \lambda_j = 0$, and the solution is $\hat{\boldsymbol{\alpha}}_j = \mathbf{z}_j - \frac{\lambda_j}{2} \geq 0$. So $\mathbf{z}_j \geq \frac{\lambda_j}{2} > 0$, and the solution $\hat{\boldsymbol{\alpha}}_j$ can be written as $\hat{\boldsymbol{\alpha}}_j = \text{sgn}(\mathbf{z}_j) * (|\mathbf{z}_j| - \frac{\lambda_j}{2})$, where sgn($\bullet$) is the sign function.

(b) If $\boldsymbol{\alpha}_j < 0$, we have $2(\boldsymbol{\alpha}_j - \mathbf{z}_j) - \lambda_j = 0$ and the solution is $\hat{\boldsymbol{\alpha}}_j = \mathbf{z}_j + \frac{\lambda_j}{2} < 0$. So $\mathbf{z}_j < -\frac{\lambda_j}{2} < 0$, and the solution $\hat{\boldsymbol{\alpha}}_j$ can be written as $\hat{\boldsymbol{\alpha}}_j = \text{sgn}(\mathbf{z}_j) * (-\mathbf{z}_j - \frac{\lambda_j}{2}) = \text{sgn}(\mathbf{z}_j) * (|\mathbf{z}_j| - \frac{\lambda_j}{2})$.

In summary, we have the final solution of the weighted sparse coding problem (8.1) as:

$$\hat{\boldsymbol{\alpha}} = \text{sgn}(\mathbf{D^T y}) \odot \max(|\mathbf{D^T y}| - \boldsymbol{\lambda}, 0), \tag{8.5}$$

where $\boldsymbol{\lambda} = \frac{1}{2}[\lambda_1, \lambda_2, ..., \lambda_{3p^2}]^\top$ is the vector of regularization parameter and $\odot$ means element-wise multiplication.

8.2 Proof of the Theorem 7

Let $\mathcal{A} \in \mathbb{R}^{(3p^2-r)\times M}, \mathcal{Y} \in \mathbb{R}^{3p^2\times M}$ be two given data matrices. Denote by $\mathcal{E} \in \mathbb{R}^{3p^2\times r}$ the external subdictionary and $\mathcal{D} \in \mathbb{R}^{3p^2\times(3p^2-r)}$ the internal subdictionary. For simplicity, we assume $3p^2 \geq M$. The problem in **Theorem 7** is as follows:

$$\begin{aligned} \hat{\mathcal{D}} = &\arg\min_{\mathcal{D}} \|\mathcal{Y} - \mathcal{D}\mathcal{A}\|_F^2 \\ &\text{s.t.} \quad \mathcal{D}^\top\mathcal{D} = \boldsymbol{I}_{(3p^2-r)\times(3p^2-r)}, \mathcal{E}^\top\mathcal{D} = \mathbf{0}_{r\times(3p^2-r)}. \end{aligned} \tag{8.6}$$

Proof. We firstly prove the necessary condition. Since $\mathcal{D}^\top\mathcal{D} = \boldsymbol{I}_{(3p^2-r)\times(3p^2-r)}$, we have

$$\begin{aligned} \hat{\mathcal{D}} = \arg\min_{\mathcal{D}} \|\mathcal{Y} - \mathcal{D}\mathcal{A}\|_F^2 = \arg\max_{\mathcal{D}} \text{Tr}(\mathcal{A}\mathcal{Y}^\top\mathcal{D}) \\ \text{s.t.} \quad \mathcal{D}^\top\mathcal{D} = \boldsymbol{I}_{(3p^2-r)\times(3p^2-r)}, \mathcal{E}^\top\mathcal{D} = \mathbf{0}_{r\times(3p^2-r)}. \end{aligned} \tag{8.7}$$

The Lagrange function is $\mathcal{L} = \text{Tr}(\mathcal{A}\mathcal{Y}^\top\mathcal{D}) - \text{Tr}(\Gamma_1(\mathcal{D}^\top\mathcal{D} - \boldsymbol{I}_{(3p^2-r)\times(3p^2-r)})) - \text{Tr}(\Gamma_2(\mathcal{D}^\top\mathcal{E}))$, where Γ_1 and Γ_2 are the Lagrange multipliers. Take the derivative of $\mathcal{L}$ w.r.t. $\mathcal{D}$ and set it to be matrix $\mathbf{0}$ of conformal dimensions, we can get

$$\partial\mathcal{L}/\partial\mathcal{D} = \mathcal{Y}\mathcal{A}^\top - \mathcal{D}(\Gamma_1 + \Gamma_1^\top) - \mathcal{E}\Gamma_2^\top = \mathbf{0}_{3p^2\times(3p^2-r)}. \tag{8.8}$$

Since $\mathcal{D}^\top\mathcal{D} = \boldsymbol{I}_{(3p^2-r)\times(3p^2-r)}$ and $\mathcal{E}^\top\mathcal{D} = \mathbf{0}_{3p^2\times(3p^2-r)}$, by left multiplying both sides of the Eq. (8.9) by $\mathcal{E}^\top$, we have

$$\mathcal{E}^\top\mathcal{Y}\mathcal{A}^\top = \Gamma_2^\top. \tag{8.9}$$

Put the Eq. (8.9) back into Eq. (8.8), we have

$$(\boldsymbol{I}_{3p^2\times3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top = \mathcal{D}(\Gamma_1 + \Gamma_1^\top). \tag{8.10}$$

Right multiplying both sides of Eq. (8.10) by $\mathcal{D}^\top$, we have

$$(\boldsymbol{I}_{3p^2\times3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top\mathcal{D}^\top = \mathcal{D}(\Gamma_1 + \Gamma_1^\top)\mathcal{D}^\top. \tag{8.11}$$

This shows that $(\boldsymbol{I}_{3p^2\times3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top\mathcal{D}^\top$ is a symmetric matrix of order $3p^2 \times 3p^2$. Then we perform economy (or reduced) singular value decomposition (SVD) [**eckart1936approximation**] on $(\boldsymbol{I}_{3p^2\times3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top = \mathcal{U}\Sigma\mathcal{V}^\top$, there is

$$(\boldsymbol{I}_{3p^2\times3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top\mathcal{D}^\top = \mathcal{U}\Sigma\mathcal{V}^\top\mathcal{D}^\top = \mathcal{D}\mathcal{V}\Sigma\mathcal{U}^\top. \tag{8.12}$$

Hence, we have $\mathcal{U} = \mathcal{D}\mathcal{V}$, or equivalently $\hat{\mathcal{D}} = \mathcal{U}\mathcal{V}^\top$. The necessary condition is proved.

Now we prove the sufficient condition. If $\hat{\mathcal{D}} = \mathcal{U}\mathcal{V}^\top$, then $\hat{\mathcal{D}}^\top\hat{\mathcal{D}} = \boldsymbol{I}_{(3p^2-r)\times(3p^2-r)}$. To prove $\mathcal{E}^\top\hat{\mathcal{D}} = \mathbf{0}_{3p^2\times(3p^2-r)}$, we left multiply both sides of Eq. (8.12) by $\mathcal{E}^\top$ and have $\mathbf{0}_{3p^2\times(3p^2-r)} = \mathcal{E}^\top(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top\hat{\mathcal{D}}^\top = \mathcal{E}^\top\mathcal{U}\Sigma\mathcal{V}^\top\hat{\mathcal{D}}^\top = \mathcal{E}^\top\mathcal{U}\Sigma\mathcal{U}^\top$. It means that $\mathcal{E}^\top\mathcal{U}\Sigma\mathcal{U}^\top = \mathbf{0}_{3p^2\times 3p^2}$. This only happens when $\mathcal{E}^\top\mathcal{U} = \mathbf{0}_{3p^2\times(3p^2-r)}$ since $\text{rank}(\Sigma) = 3p^2 - r$ and $\mathcal{U}\Sigma\mathcal{U}^\top$ is positive definite. Then $\mathcal{E}^\top\hat{\mathcal{D}} = \mathcal{E}^\top\mathcal{U}\mathcal{V}^\top = \mathbf{0}_{3p^2\times(3p^2-r)}$.

Finally we prove that $\hat{\mathcal{D}} = \mathcal{U}\mathcal{V}^\top$ is the solution of

$$\hat{\mathcal{D}} = \arg\min_{\mathcal{D}} \|\mathcal{Y} - \mathcal{D}\mathcal{A}\|_F^2 = \arg\max_{\mathcal{D}} \text{Tr}(\mathcal{Y}^\top\mathcal{D}\mathcal{A}). \tag{8.13}$$

Note that by cyclic perturbation which retains the trace unchanged and due to $\mathcal{E}^\top\hat{\mathcal{D}} = \mathbf{0}_{3p^2\times(3p^2-r)}$, we have $\text{Tr}(\mathcal{Y}^\top\hat{\mathcal{D}}\mathcal{A}) = \text{Tr}(\mathcal{Y}\mathcal{A}^\top\hat{\mathcal{D}}^\top) = \text{Tr}((\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top\hat{\mathcal{D}}^\top) = \text{Tr}(\mathcal{U}\Sigma\mathcal{V}^\top\mathcal{V}\mathcal{U}^\top) = \text{Tr}(\Sigma)$. For every $\mathcal{D}$ satisfying that $\mathcal{D}^\top\mathcal{D} = \boldsymbol{I}_{(3p^2-r)\times(3p^2-r)}$, $\mathcal{E}^\top\mathcal{D} = \mathbf{0}_{3p^2\times(3p^2-r)}$, we have $\text{Tr}(\mathcal{Y}^\top\mathcal{D}\mathcal{A}) = \text{Tr}((\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top\mathcal{D}^\top) = \text{Tr}(\mathcal{U}\Sigma\mathcal{V}^\top\mathcal{D}^\top) = \text{Tr}(\Sigma\mathcal{V}^\top\mathcal{D}^\top\mathcal{U})$. By using a generalization version [**TenBerge1983**] of the Kristof's Theorem [**Kristof1970515**], we have $\text{Tr}(\mathcal{Y}^\top\mathcal{D}\mathcal{A}) = \text{Tr}(\Sigma\mathcal{V}^\top\mathcal{D}^\top\mathcal{U}) \leq \text{Tr}(\Sigma)$. The equality is obtained at $\mathcal{V}^\top\mathcal{D}^\top\mathcal{U} = \boldsymbol{I}_{(3p^2-r)\times(3p^2-r)}$, i.e., $\mathcal{D} = \mathcal{U}\mathcal{V}^\top = \hat{\mathcal{D}}$. This completes the proof. □

8.3 Proof of the Theorem 8

Before we prove the Theorem 8, we need firstly prove the following Lemma 1.

Lemma 1: Let $\mathcal{E} \in \mathbb{R}^{3p^2\times r}$ be an orthogonal matrix with $\mathcal{E}^\top\mathcal{E} = \boldsymbol{I}_{r\times r}$, then $\text{rank}(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top) \geq 3p^2 - r$.

Proof. Since $\text{rank}(\mathcal{E}\mathcal{E}^\top) \leq \min\{\text{rank}(\mathcal{E}), \text{rank}(\mathcal{E}^\top)\} = r$ and $\text{rank}(\mathcal{E}\mathcal{E}^\top) \geq \text{rank}(\mathcal{E}) + \text{rank}(\mathcal{E}^\top) - r = 2r - r = r$ by Sylvester's inequality, we have $\text{rank}(\mathcal{E}\mathcal{E}^\top) = r$. Then, $\text{rank}(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top) \geq \text{rank}(\boldsymbol{I}_{3p^2\times 3p^2}) - \text{rank}(\mathcal{E}\mathcal{E}^\top) \geq 3p^2 - r$. □

The rank(Σ) (Σ is defined in Theorem 7) depends on rank($\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top$), rank($\mathcal{Y}$) and rank($\mathcal{A}$). Note that rank($\mathcal{Y}$) $\geq M$ and rank($\mathcal{A}$) $\geq \min\{3p^2, M\}$ and rank($\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top$) $\geq 3p^2 - r$. Hence, rank(Σ) $\leq \min\{3p^2 - r, M\}$.

Now we prove the Theorem 8:

Proof. a) If $(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top \in \mathbb{R}^{3p^2\times(3p^2-r)}$ is nonsingular, i.e., rank(Σ) = $3p^2 - r$, Σ may have distinct or multiple non-zero singular values. In the SVD [**eckart1936approximation**] of $(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top = \mathcal{U}\Sigma\mathcal{V}^\top$, the singular vectors in $\mathcal{U}$ and $\mathcal{V}$ can be determined up to orientation. Hence, we can reformulate it as

$$(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top = \mathcal{U}^*\mathcal{K}_u\Sigma\mathcal{K}_v(\mathcal{V}^*)^\top, \tag{8.14}$$

where $\mathcal{U}^* \in \mathbb{R}^{3p^2\times(3p^2-r)}$ and $\mathcal{V}^* \in \mathbb{R}^{(3p^2-r)\times(3p^2-r)}$ are arbitrarily orientated singular vectors of $\mathcal{U}$ and $\mathcal{V}$, respectively. The $\mathcal{K}_u$ and $\mathcal{K}_v$ are diagonal matrices with $+1$ or -1 as diagonal elements in arbitrary distribution. $\Sigma \in \mathbb{R}^{(3p^2-r)\times(3p^2-r)}$ is a diagonal matrix with singular values in non-increasing order, i.e., $\Sigma_{11} \geq \Sigma_{22} \geq \ldots \geq \Sigma_{(3p^2-r)(3p^2-r)} \geq 0$. If we fix $\mathcal{K}_u$, then $\mathcal{K}_v$ is uniquely determined to meet the above requirements of Σ. If the orientations of the singular vectors of $\mathcal{U}^*$ are fixed, then $\mathcal{U} = \mathcal{U}^*\mathcal{K}_u$ is determined, so do the orientations of the singular vectors of $\mathcal{V}^*$ and $\mathcal{V}^\top = \mathcal{K}_v(\mathcal{V}^*)^\top$. In this case, the solution of $\hat{\mathcal{D}} = \mathcal{U}\mathcal{V}^\top = \mathcal{U}^*\mathcal{K}_u\mathcal{K}_v(\mathcal{V}^*)^\top$ is unique. When Σ has multiple singular values, the unique solution of $\hat{\mathcal{D}}$ can be proved in a similar way.

b) If $(\boldsymbol{I}_{3p^2\times 3p^2} - \mathcal{E}\mathcal{E}^\top)\mathcal{Y}\mathcal{A}^\top$ is singular, i.e., $0 \leq$ rank(Σ) $< 3p^2 - r$, and Σ has $3p^2 - r -$ rank(Σ) (at least one) zero singular values. The discussion in a) can still be applied to the singular vectors corresponding to the nonzero singular values, and the production of these singular vectors in $\mathcal{U}$ and $\mathcal{V}$ is still unique. However, the singular vectors corresponding to the zero singular values could be in arbitrary orientations as long as they satisfy the conditions of $\mathcal{U}^\top\mathcal{U} = \mathcal{V}^\top\mathcal{V} = \mathcal{V}\mathcal{V}^\top = \boldsymbol{I}_{(3p^2-r)\times(3p^2-r)}$. Since $\mathcal{U} \in \mathbb{R}^{3p^2\times(3p^2-r)}$, $\mathcal{U}\mathcal{U}^\top$ no longer equals to the identity matrix of order $3p^2 \times 3p^2$. From Eq. (8.12), we have

$$\mathcal{U}\Sigma\mathcal{V}^\top\mathcal{D}^\top = \mathcal{D}\mathcal{V}\Sigma\mathcal{U}^\top \tag{8.15}$$

Right multiplying both sides of Eq. (8.15) by $\mathcal{D}\mathcal{V}$ and left multiplying each side by $\mathcal{U}^\top$, we have

$$\Sigma = \mathcal{U}^\top\mathcal{D}\mathcal{V}\Sigma\mathcal{U}^\top\mathcal{D}\mathcal{V} \tag{8.16}$$

Hence, $\Delta = \mathcal{U}^\top\mathcal{D}\mathcal{V} \in \mathbb{R}^{(3p^2-r)\times(3p^2-r)}$ is a diagonal matrix, the diagonal elements of which are

$$\Delta_{ii} = \begin{cases} 1 & \text{if } 1 \le i \le \text{rank}(\Sigma); \\ \pm 1 & \text{if } \text{rank}(\Sigma) < i \le 3p^2 - r. \end{cases}$$

Thus, we have $\mathcal{D} = \mathcal{U}\Delta\mathcal{V}^\top$. That is, if $\text{rank}(\Sigma) < 3p^2 - r$, once we get the solution of $\hat{\mathcal{D}} = \mathcal{U}\mathcal{V}^\top$ in problem (8.6), $\mathcal{D} = \mathcal{U}\Delta\mathcal{V}^\top$ with suitable Δ is also the solution of problem (8.6). In fact, the number of solutions $\hat{\mathcal{D}}$ for problem (8.6) is $2^{3p^2-r-\text{rank}(\Sigma)}$ given fixed $\mathcal{U}$ and $\mathcal{V}$. □

8.4 Proof of Theorem 1.

Theorem 6. *Assume that the weights in $\boldsymbol{w}$ are in a non-descending order, the sequence $\{\boldsymbol{X}_k\}$, $\{\boldsymbol{Z}_k\}$, and $\{\boldsymbol{A}_k\}$ generated in Algorithm 1 satisfy:*

$$(a)\ \lim_{k\to\infty} \|\boldsymbol{X}_{k+1} - \boldsymbol{Z}_{k+1}\|_F = 0; \quad (b)\ \lim_{k\to\infty} \|\boldsymbol{X}_{k+1} - \boldsymbol{X}_k\|_F = 0; \quad (c)\ \lim_{k\to\infty} \|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F = 0. \tag{8.17}$$

Proof. 1. Firstly, we prove that the sequence $\{\boldsymbol{A}_k\}$ generated by Algorithm 1 is upper bounded. Let $\boldsymbol{X}_{k+1} + \rho_k^{-1}\boldsymbol{A}_k = \boldsymbol{U}_k\boldsymbol{\Sigma}_k\boldsymbol{V}_k^\top$ be its singular value decomposition (SVD) [**eckart1936approximation**] in the $(k+1)$-th iteration. According to Corollary 1 of [**wnnmijcv**], we can have the SVD of $\boldsymbol{Z}_{k+1}$ as $\boldsymbol{Z}_{k+1} = \boldsymbol{U}_k\hat{\boldsymbol{\Sigma}}_k\boldsymbol{V}_k^\top = \boldsymbol{U}_k\mathcal{S}_{\frac{\boldsymbol{w}}{\rho_k}}(\boldsymbol{\Sigma}_k)\boldsymbol{V}_k^\top$. Then we have

$$\|\boldsymbol{A}_{k+1}\|_F = \|\boldsymbol{A}_k + \rho_k(\boldsymbol{X}_{k+1} - \boldsymbol{Z}_{k+1})\|_F = \rho_k\|\rho_k^{-1}\boldsymbol{A}_k + \boldsymbol{X}_{k+1} - \boldsymbol{Z}_{k+1}\|_F \tag{8.18}$$

$$= \rho_k\|\boldsymbol{U}_k\boldsymbol{\Sigma}_k\boldsymbol{V}_k^\top - \boldsymbol{U}_k\mathcal{S}_{\frac{\boldsymbol{w}}{\rho_k}}(\boldsymbol{\Sigma}_k)\boldsymbol{V}_k^\top\|_F = \rho_k\|\boldsymbol{\Sigma}_k - \mathcal{S}_{\frac{\boldsymbol{w}}{\rho_k}}(\boldsymbol{\Sigma}_k)\|_F \tag{8.19}$$

$$= \rho_k\sqrt{\sum_i(\boldsymbol{\Sigma}_k^{ii} - \mathcal{S}_{\frac{w_i}{\rho_k}}(\boldsymbol{\Sigma}_k^{ii}))^2} \le \rho_k\sqrt{\sum_i(\frac{w_i}{\rho_k})^2} = \sqrt{\sum_i w_i^2}. \tag{8.20}$$

The inequality in the second last step can be proved as follows: given the diagonal matrix $\boldsymbol{\Sigma}_k$, we define $\boldsymbol{\Sigma}_k^{ii}$ as the i-th element of $\boldsymbol{\Sigma}_k$. If $\boldsymbol{\Sigma}_k^{ii} \ge \frac{w_i}{\rho_k}$, we have $\mathcal{S}_{\frac{w_i}{\rho_k}}(\boldsymbol{\Sigma}_k^{ii}) = \boldsymbol{\Sigma}_k^{ii} - \frac{w_i}{\rho_k}$. If $\boldsymbol{\Sigma}_k^{ii} < \frac{w_i}{\rho_k}$, we have $\mathcal{S}_{\frac{w_i}{\rho_k}}(\boldsymbol{\Sigma}_k^{ii}) = 0$. Overall, we have $|\boldsymbol{\Sigma}_k^{ii} - \mathcal{S}_{\frac{w_i}{\rho_k}}(\boldsymbol{\Sigma}_k^{ii})| \le \frac{w_i}{\rho_k}$ and hence the inequality holds. Hence, the sequence $\{\boldsymbol{A}_k\}$ is upper bounded.

2. Secondly, we prove that the sequence of Lagrangian function $\{\mathcal{L}(\boldsymbol{X}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{A}_k, \rho_k)\}$ is also upper bounded. Since we have the globally optimal solution of $\boldsymbol{X}$ and $\boldsymbol{Z}$ in their corresponding subproblems, we always have

$$\mathcal{L}(\boldsymbol{X}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{A}_k, \rho_k) \leq \mathcal{L}(\boldsymbol{X}_k, \boldsymbol{Z}_k, \boldsymbol{A}_k, \rho_k). \tag{8.21}$$

Based on the updating rule that $\boldsymbol{A}_{k+1} = \boldsymbol{A}_k + \rho_k(\boldsymbol{X}_{k+1} - \boldsymbol{Z}_{k+1})$, we have

$$\mathcal{L}(\boldsymbol{X}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{A}_{k+1}, \rho_{k+1}) \tag{8.22}$$

$$= \mathcal{L}(\boldsymbol{X}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{A}_k, \rho_k) + \langle \boldsymbol{A}_{k+1} - \boldsymbol{A}_k, \boldsymbol{X}_{k+1} - \boldsymbol{Z}_{k+1} \rangle + \frac{\rho_{k+1} - \rho_k}{2} \|\boldsymbol{X}_{k+1} - \boldsymbol{Z}_{k+1}\|_F^2 \tag{8.23}$$

$$= \mathcal{L}(\boldsymbol{X}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{A}_k, \rho_k) + \frac{\rho_{k+1} + \rho_k}{2\rho_k^2} \|\boldsymbol{A}_{k+1} - \boldsymbol{A}_k\|_F^2. \tag{8.24}$$

Since the sequence $\{\boldsymbol{A}_k\}$ is upper bounded, the sequence $\{\boldsymbol{A}_{k+1} - \boldsymbol{A}_k\}$ is also upper bounded. Denote by a the upper bound of $\{\boldsymbol{A}_{k+1} - \boldsymbol{A}_k\}$, i.e., $\|\boldsymbol{A}_{k+1} - \boldsymbol{A}_k\|_F \leq a$ holds for $\forall k \geq 0$, we have

$$\mathcal{L}(\boldsymbol{X}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{A}_{k+1}, \rho_{k+1}) \leq \mathcal{L}(\boldsymbol{X}_1, \boldsymbol{Z}_1, \boldsymbol{A}_0, \rho_0) + a^2 \sum_{k=0}^{\infty} \frac{\rho_{k+1} + \rho_k}{2\rho_k^2} \tag{8.25}$$

$$= \mathcal{L}(\boldsymbol{X}_1, \boldsymbol{Z}_1, \boldsymbol{A}_0, \rho_0) + a^2 \sum_{k=0}^{\infty} \frac{\mu + 1}{2\mu^k \rho_0} \tag{8.26}$$

$$\leq \mathcal{L}(\boldsymbol{X}_1, \boldsymbol{Z}_1, \boldsymbol{A}_0, \rho_0) + \frac{a^2}{\rho_0} \sum_{k=0}^{\infty} \frac{1}{\mu^{k-1}}. \tag{8.27}$$

The last inequality holds since $\mu > 1$ and $\mu + 1 < 2\mu$. Therefore, we have $\sum_{k=0}^{\infty} \frac{1}{\mu^{k-1}} < \infty$ and the sequence of Lagrangian function $\{\mathcal{L}(\boldsymbol{X}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{A}_{k+1}, \rho_{k+1})\}$ is upper bounded.

3. Thirdly, we prove that the sequences of $\{\boldsymbol{X}_k\}$ and $\{\boldsymbol{Z}_k\}$ are upper bounded. Since

$$\|\boldsymbol{W}(\boldsymbol{Y} - \boldsymbol{X}_k)\|_F^2 + \|\boldsymbol{Z}_k\|_{\boldsymbol{w},*} = \mathcal{L}(\boldsymbol{X}_k, \boldsymbol{Z}_k, \boldsymbol{A}_{k-1}, \rho_{k-1}) - \langle \boldsymbol{A}_{k-1}, \boldsymbol{X}_k - \boldsymbol{Z}_k \rangle - \frac{\rho_{k-1}}{2} \|\boldsymbol{X}_k - \boldsymbol{Z}_k\|_F^2 \tag{8.28}$$

$$= \mathcal{L}(\boldsymbol{X}_k, \boldsymbol{Z}_k, \boldsymbol{A}_{k-1}, \rho_{k-1}) + \frac{1}{2\rho_{k-1}} (\|\boldsymbol{A}_{k-1}\|_F^2 - \|\boldsymbol{A}_k\|_F^2), \tag{8.29}$$

both $\{\boldsymbol{W}(\boldsymbol{Y}-\boldsymbol{X}_k)\}$ and $\{\boldsymbol{Z}_k\}$ are upper bounded, and hence the sequence $\{\boldsymbol{X}_k\}$ is bounded by the Cauchy-Schwarz inequality and triangle inequality. We can obtain that

$$\lim_{k\to\infty}\|\boldsymbol{X}_{k+1}-\boldsymbol{Z}_{k+1}\|_F=\lim_{k\to\infty}\rho_k^{-1}\|\boldsymbol{A}_{k+1}-\boldsymbol{A}_k\|_F=0, \tag{8.30}$$

and the equation (a) is proved.

4. Then we can prove that

$$\lim_{k\to\infty}\|\boldsymbol{X}_{k+1}-\boldsymbol{X}_k\|_F=\lim_{k\to\infty}\|(\boldsymbol{W}^\top\boldsymbol{W}+\frac{\rho_k}{2}\boldsymbol{I})^{-1}(\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{Y}-\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{Z}_k-\frac{1}{2}\boldsymbol{A}_k)-\rho_{k-1}^{-1}(\boldsymbol{A}_k- \tag{8.31}$$

$$\leq\lim_{k\to\infty}(\|(\boldsymbol{W}^\top\boldsymbol{W}+\frac{\rho_k}{2}\boldsymbol{I})^{-1}(\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{Y}-\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{Z}_k-\frac{1}{2}\boldsymbol{A}_k)\|_F+\rho_{k-1}^{-1}\|\boldsymbol{A}_k \tag{8.32}$$

$$=0, \tag{8.33}$$

and hence the equation (b) is proved.

5. Finally, the equation (c) can be proved by checking that

$$\lim_{k\to\infty}\|\boldsymbol{Z}_{k+1}-\boldsymbol{Z}_k\|_F=\lim_{k\to\infty}\|\boldsymbol{X}_k+\rho_{k-1}^{-1}\boldsymbol{A}_{k-1}-\boldsymbol{Z}_k+\boldsymbol{X}_{k+1}-\boldsymbol{X}_k-\rho_{k-1}^{-1}\boldsymbol{A}_{k-1}+\rho_k^{-1}\boldsymbol{A}_k-\rho_k^{-1}\boldsymbol{A} \tag{8.34}$$

$$\leq\lim_{k\to\infty}(\|\boldsymbol{\Sigma}_{k-1}-\mathcal{S}_{\boldsymbol{w}/\rho_{k-1}}(\boldsymbol{\Sigma}_{k-1})\|_F+\|\boldsymbol{X}_{k+1}-\boldsymbol{X}_k\|_F+\|\rho_{k-1}^{-1}\boldsymbol{A}_{k-1}+\rho_k^{-1}\boldsymbol{A} \tag{8.35}$$

$$\leq\lim_{k\to\infty}(\rho_{k-1}^{-1}\|\boldsymbol{w}\|_F+\|\boldsymbol{X}_{k+1}-\boldsymbol{X}_k\|_F+\|\rho_{k-1}^{-1}\boldsymbol{A}_{k-1}+\rho_k^{-1}\boldsymbol{A}_{k+1}-\rho_k^{-1}\boldsymbol{A}_k\|_F) \tag{8.36}$$

$$=0, \tag{8.37}$$

where $\boldsymbol{U}_{k-1}\boldsymbol{\Sigma}_{k-1}\boldsymbol{V}_{k-1}^\top$ is the SVD of the matrix $\boldsymbol{X}_k+\rho_{k-1}^{-1}\boldsymbol{A}_{k-1}$. □

8.5 Proofs of Theorem 1 and Theorem 2

Theorem 7. *Given $\rho_{k+1} = \mu\rho_k$ ($\rho_0 > 0$) for $k \geq 0$ and $\mu > 1$, the sequences $\{\boldsymbol{C}_k\}$ and $\{\boldsymbol{Z}_k\}$ generated in Algorithm 1 satisfy:*

$$(a)\ \|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F = \mathcal{O}(\rho_k^{-1})\ \textit{as}\ k \to +\infty,\ \textit{i.e.},\ \lim_{k\to+\infty} \|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F = 0; \tag{8.38}$$

$$(b)\ \textit{If}\ \|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F = \mathcal{O}(\rho_k^{-1})\ \textit{as}\ k \to +\infty,\ \textit{then}\ \lim_{k\to+\infty} \|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F = 0; \tag{8.39}$$

$$(c)\ \textit{If}\ \lim_{k\to+\infty} \|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F = 0,\ \textit{then}\ \lim_{k\to+\infty} \|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F = 0. \tag{8.40}$$

Proof.
1. Firstly, we prove that the sequence $\{\boldsymbol{\Delta}_k\}$ generated by Algorithm 1 is upper bounded. Denote by $\boldsymbol{T}_k = \boldsymbol{C}_{k+1} + \rho_k^{-1}\boldsymbol{\Delta}_k$ in the $(k+1)$-th iteration. According to Eq. (16) in the main paper, we have $\boldsymbol{Z}_{k+1} = \mathcal{S}_{\frac{1}{2\rho_k}}(\boldsymbol{T}_k)$. Then we have

$$\|\boldsymbol{\Delta}_{k+1}\|_F = \|\boldsymbol{\Delta}_k + \rho_k(\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1})\|_F = \rho_k\|\rho_k^{-1}\boldsymbol{\Delta}_k + \boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F \tag{8.41}$$

$$= \rho_k\|\boldsymbol{T}_k - \mathcal{S}_{\rho_k^{-1}}(\boldsymbol{T}_k)\|_F = \rho_k\sqrt{\sum_{i,j}(\boldsymbol{T}_k^{ij} - \mathcal{S}_{\rho_k^{-1}}(\boldsymbol{T}_k^{ij}))^2} \tag{8.42}$$

$$\leq \rho_k\sqrt{\sum_{i,j}\rho_k^{-2}} = 3p^2M. \tag{8.43}$$

The inequality in Eq. (6) can be proved as follows: given the matrix $\boldsymbol{T}_k$, we define $\boldsymbol{T}_k^{ij}$ as the element of the i-th row and j-th column of $\boldsymbol{T}_k$. If $\boldsymbol{T}_k^{ij} \geq \rho_k^{-1}$, we have $\mathcal{S}_{\rho_k^{-1}}(\boldsymbol{T}_k^{ij}) = \boldsymbol{T}_k^{ij} - \rho_k^{-1}$. If $\boldsymbol{T}_k^{ij} < \rho_k^{-1}$, we have $\mathcal{S}_{\rho_k^{-1}}(\boldsymbol{T}_k^{ij}) = 0$. Overall, we have $|\boldsymbol{T}_k^{ij} - \mathcal{S}_{\rho_k^{-1}}(\boldsymbol{T}_k^{ij})| \leq \rho_k^{-1}$ and the inequality holds. Hence, the sequence $\{\boldsymbol{\Delta}_k\}$ is upper bounded.

2. Secondly, we prove that the sequence of Lagrangian function $\{\mathcal{L}(\boldsymbol{C}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{\Delta}_k, \rho_k)\}$ is also upper bounded. Since we have the globally optimal solution of $\boldsymbol{C}$ and $\boldsymbol{Z}$ in their corresponding subproblems, we always have

$$\mathcal{L}(\boldsymbol{C}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{\Delta}_k, \rho_k) \leq \mathcal{L}(\boldsymbol{C}_k, \boldsymbol{Z}_k, \boldsymbol{\Delta}_k, \rho_k). \tag{8.44}$$

Based on the updating rule that $\boldsymbol{\Delta}_{k+1} = \boldsymbol{\Delta}_k + \rho_k(\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1})$, we have

$$\mathcal{L}(\boldsymbol{C}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{\Delta}_{k+1}, \rho_{k+1}) \tag{8.45}$$

$$= \mathcal{L}(\boldsymbol{C}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{\Delta}_k, \rho_k) + \langle \boldsymbol{\Delta}_{k+1} - \boldsymbol{\Delta}_k, \boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1} \rangle + \frac{\rho_{k+1} - \rho_k}{2} \|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F^2 \tag{8.46}$$

$$= \mathcal{L}(\boldsymbol{C}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{\Delta}_k, \rho_k) + \frac{\rho_{k+1} + \rho_k}{2\rho_k^2} \|\boldsymbol{\Delta}_{k+1} - \boldsymbol{\Delta}_k\|_F^2. \tag{8.47}$$

Since the sequence $\{\boldsymbol{\Delta}_k\}$ is upper bounded, the sequence $\{\boldsymbol{\Delta}_{k+1} - \boldsymbol{\Delta}_k\}$ is also upper bounded. Denote by a the upper bound of $\{\boldsymbol{\Delta}_{k+1} - \boldsymbol{\Delta}_k\}$, i.e., $\|\boldsymbol{\Delta}_{k+1} - \boldsymbol{\Delta}_k\|_F \leq a$ holds for $\forall k \geq 0$, we have

$$\mathcal{L}(\boldsymbol{C}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{\Delta}_{k+1}, \rho_{k+1}) \leq \mathcal{L}(\boldsymbol{C}_1, \boldsymbol{Z}_1, \boldsymbol{A}_0, \rho_0) + a^2 \sum_{k=0}^{+\infty} \frac{\rho_{k+1} + \rho_k}{2\rho_k^2} \tag{8.48}$$

$$= \mathcal{L}(\boldsymbol{C}_1, \boldsymbol{Z}_1, \boldsymbol{A}_0, \rho_0) + a^2 \sum_{k=0}^{+\infty} \frac{\mu + 1}{2\mu^k \rho_0} \tag{8.49}$$

$$\leq \mathcal{L}(\boldsymbol{C}_1, \boldsymbol{Z}_1, \boldsymbol{A}_0, \rho_0) + \frac{a^2}{\rho_0} \sum_{k=0}^{+\infty} \frac{1}{\mu^{k-1}}. \tag{8.50}$$

The last inequality holds since $\mu > 1$ and $\mu + 1 < 2\mu$. Therefore, we have $\sum_{k=0}^{+\infty} \frac{1}{\mu^{k-1}} < +\infty$ and the sequence of Lagrangian function $\{\mathcal{L}(\boldsymbol{C}_{k+1}, \boldsymbol{Z}_{k+1}, \boldsymbol{\Delta}_{k+1}, \rho_{k+1})\}$ is upper bounded.

3. Thirdly, we prove that the sequences of $\{\boldsymbol{C}_k\}$ and $\{\boldsymbol{Z}_k\}$ are upper bounded. Since

$$\|\boldsymbol{W}_1(\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{W}_3\boldsymbol{C}_k)\boldsymbol{W}_2\|_F^2 + \|\boldsymbol{Z}_k\|_1 \tag{8.51}$$

$$= \mathcal{L}(\boldsymbol{C}_k, \boldsymbol{Z}_k, \boldsymbol{\Delta}_{k-1}, \rho_{k-1}) - \langle \boldsymbol{\Delta}_{k-1}, \boldsymbol{C}_k - \boldsymbol{Z}_k \rangle - \frac{\rho_{k-1}}{2} \|\boldsymbol{C}_k - \boldsymbol{Z}_k\|_F^2 \tag{8.52}$$

$$= \mathcal{L}(\boldsymbol{C}_k, \boldsymbol{Z}_k, \boldsymbol{\Delta}_{k-1}, \rho_{k-1}) + \frac{1}{2\rho_{k-1}} (\|\boldsymbol{\Delta}_{k-1}\|_F^2 - \|\boldsymbol{\Delta}_k\|_F^2), \tag{8.53}$$

both $\{\boldsymbol{W}_1(\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{W}_3\boldsymbol{C}_k)\boldsymbol{W}_2\}$ and $\{\boldsymbol{Z}_k\}$ are upper bounded, and hence the sequence $\{\boldsymbol{C}_k\}$ is bounded by the Cauchy-Schwarz inequality and triangle inequality. We can obtain that

$$\lim_{k \to +\infty} \|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F = \lim_{k \to +\infty} \rho_k^{-1} \|\boldsymbol{\Delta}_{k+1} - \boldsymbol{\Delta}_k\|_F = 0. \tag{8.54}$$

Since the sequence of $\{\boldsymbol{\Delta}_k\}$ is upper bounded, we can also easily obtain that $\|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F = \mathcal{O}(\rho_k^{-1})$ as $k \to +\infty$ and the equation (a) is proved.

4. Now we prove the equation (b). Denote by $\boldsymbol{A} = \boldsymbol{W}_3^\top \boldsymbol{D}^\top \boldsymbol{W}_1^\top \boldsymbol{W}_1 \boldsymbol{D} \boldsymbol{W}_3$, $\boldsymbol{B} = \frac{1}{2}(\boldsymbol{W}_2 \boldsymbol{W}_2^\top)^{-1}$, and $\boldsymbol{E} = \boldsymbol{W}_3^\top \boldsymbol{D}^\top \boldsymbol{W}_1^\top \boldsymbol{W}_1 \boldsymbol{Y}$, and $\boldsymbol{F}_k = \rho_k(\boldsymbol{Z}_k - \frac{1}{\rho_k}\boldsymbol{\Delta}_k)\boldsymbol{B}$. According to the Eq. (13) in the main paper, we have

$$\boldsymbol{A}\boldsymbol{C}_{k+1} + \rho_k \boldsymbol{C}_{k+1}\boldsymbol{B} = \boldsymbol{E} + \boldsymbol{F}_k, \tag{8.55}$$

$$\boldsymbol{A}\boldsymbol{C}_{k} + \rho_{k-1} \boldsymbol{C}_{k}\boldsymbol{B} = \boldsymbol{E} + \boldsymbol{F}_{k-1}. \tag{8.56}$$

By calculating the difference between the Eq. (18) and Eq. (19), we have

$$\begin{aligned}
&\boldsymbol{A}(\boldsymbol{C}_{k+1} - \boldsymbol{C}_k) + \rho_k(\boldsymbol{C}_{k+1} - \boldsymbol{C}_k)\boldsymbol{B} \\
=&\boldsymbol{F}_k - \boldsymbol{F}_{k-1} + \rho_{k-1}\boldsymbol{C}_k\boldsymbol{B} - \rho_k \boldsymbol{C}_k \boldsymbol{B} \\
=&\rho_k(\boldsymbol{Z}_k - \boldsymbol{C}_k)\boldsymbol{B} + (\boldsymbol{\Delta}_{k-1} - \boldsymbol{\Delta}_k)\boldsymbol{B} - \rho_{k-1}(\boldsymbol{Z}_{k-1} - \boldsymbol{C}_k)\boldsymbol{B} \\
=&\rho_k(\boldsymbol{Z}_k - \boldsymbol{C}_k)\boldsymbol{B} + (\boldsymbol{\Delta}_{k-1} - \boldsymbol{\Delta}_k)\boldsymbol{B} + \rho_{k-1}(\boldsymbol{C}_k - \boldsymbol{Z}_k)\boldsymbol{B} + \rho_{k-1}(\boldsymbol{Z}_k - \boldsymbol{Z}_{k-1})\boldsymbol{B} \\
=&\mu(\boldsymbol{\Delta}_{k-1} - \boldsymbol{\Delta}_k)\boldsymbol{B} + \rho_{k-1}(\boldsymbol{Z}_k - \boldsymbol{Z}_{k-1})\boldsymbol{B}.
\end{aligned} \tag{8.57}$$

This equation can be transformed into the following Sylvester equation

$$(\boldsymbol{I}_n \otimes \mathbf{A} + \rho_k \boldsymbol{B}^\top \otimes \mathbf{I}_m)\text{vec}(\boldsymbol{C}_{k+1} - \boldsymbol{C}_k) = \text{vec}(\mu(\boldsymbol{\Delta}_{k-1} - \boldsymbol{\Delta}_k)\boldsymbol{B} + \rho_{k-1}(\boldsymbol{Z}_k - \boldsymbol{Z}_{k-1})\boldsymbol{B}), \tag{8.58}$$

Since $\|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F = \mathcal{O}(\rho_k^{-1})$ as $k \to +\infty$, there exist constants $N \in \mathbb{Z}$ and $c \in \mathbb{R}_+$ such that,

$$\rho_k \|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F \leq c \quad \text{for all} \quad k > N. \tag{8.59}$$

Note that $\lim_{k \to +\infty} \rho_k = 0$ and the solution of the above Sylvester equation is

$$\text{vec}(\boldsymbol{C}_{k+1} - \boldsymbol{C}_k) = (\boldsymbol{I}_n \otimes \mathbf{A} + \rho_k \boldsymbol{B}^\top \otimes \mathbf{I}_m)^{-1}\text{vec}(\mu(\boldsymbol{\Delta}_{k-1} - \boldsymbol{\Delta}_k)\boldsymbol{B} + \rho_{k-1}(\boldsymbol{Z}_k - \boldsymbol{Z}_{k-1})\boldsymbol{B}). \tag{8.60}$$

By Cauchy-Schwarz inequality, we can obtain that

$$\|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F = \|\text{vec}(\boldsymbol{C}_{k+1} - \boldsymbol{C}_k)\|_2 \tag{8.61}$$

$$= \|(\boldsymbol{I}_n \otimes \mathbf{A} + \rho_k \boldsymbol{B}^\top \otimes \mathbf{I}_m)^{-1}\text{vec}(\mu(\boldsymbol{\Delta}_{k-1} - \boldsymbol{\Delta}_k)\boldsymbol{B} + \rho_{k-1}(\boldsymbol{Z}_k - \boldsymbol{Z}_{k-1})\boldsymbol{B})\|_2 \tag{8.62}$$

$$\leq (\mu a + c)\|(\boldsymbol{I}_n \otimes \mathbf{A} + \rho_k \boldsymbol{B}^\top \otimes \mathbf{I}_m)^{-1}\|_F \|\boldsymbol{B}\|_F. \tag{8.63}$$

Since $\lim_{k\to+\infty}\rho_k = +\infty$, we have that $\lim_{k\to+\infty}\|(\boldsymbol{I}_n\otimes\mathbf{A}+\rho_k\boldsymbol{B}^\top\otimes\mathbf{I}_m)^{-1}\|_F = 0$ and

$$\lim_{k\to+\infty}\|\boldsymbol{C}_{k+1}-\boldsymbol{C}_k\|_F = 0. \tag{8.64}$$

The equation (b) is proved.

5. Now we prove the argument (c). Denote by $\boldsymbol{T}_{k-1} = \boldsymbol{C}_k + \rho_{k-1}^{-1}\boldsymbol{\Delta}_{k-1}$. Since $\lim_{k\to+\infty}\rho_k = +\infty$ and $\lim_{k\to+\infty}\|\boldsymbol{C}_{k+1}-\boldsymbol{C}_k\|_F = 0$, the argument (c) can be proved by checking that

$$\lim_{k\to+\infty}\|\boldsymbol{Z}_{k+1}-\boldsymbol{Z}_k\|_F \tag{8.65}$$

$$= \lim_{k\to+\infty}\|\boldsymbol{C}_k+\rho_{k-1}^{-1}\boldsymbol{\Delta}_{k-1}-\boldsymbol{Z}_k+\boldsymbol{C}_{k+1}-\boldsymbol{C}_k-\rho_{k-1}^{-1}\boldsymbol{\Delta}_{k-1}+\rho_k^{-1}\boldsymbol{\Delta}_k-\rho_k^{-1}\boldsymbol{\Delta}_{k+1}\|_F \tag{8.66}$$

$$\leq \lim_{k\to+\infty}(\|\boldsymbol{T}_{k-1}-\mathcal{S}_{\rho_{k-1}^{-1}}(\boldsymbol{T}_{k-1})\|_F+\|\boldsymbol{C}_{k+1}-\boldsymbol{C}_k\|_F+\|\rho_{k-1}^{-1}\boldsymbol{\Delta}_{k-1}+\rho_k^{-1}\boldsymbol{\Delta}_{k+1}-\rho_k^{-1}\boldsymbol{\Delta}_k\|_F) \tag{8.67}$$

$$\leq \lim_{k\to+\infty}(3p^2M\rho_{k-1}^{-1}+\|\boldsymbol{C}_{k+1}-\boldsymbol{C}_k\|_F+\|\rho_{k-1}^{-1}\boldsymbol{\Delta}_{k-1}+\rho_k^{-1}\boldsymbol{\Delta}_{k+1}-\rho_k^{-1}\boldsymbol{\Delta}_k\|_F) = 0. \tag{8.68}$$

That is the end of the proof. □

Theorem 8. *Assume that $\boldsymbol{A}\in\mathbb{R}^{3p^2\times 3p^2}$, $\boldsymbol{B}\in\mathbb{R}^{M\times M}$ are both symmetric and positive semi-definite matrices. If at least one of $\boldsymbol{A},\boldsymbol{B}$ is positive definite, the Sylvester equation $\boldsymbol{AC}+\boldsymbol{CB}=\boldsymbol{E}$ has a unique solution for $\boldsymbol{C}\in\mathbb{R}^{3p^2\times M}$.*

Proof. Since $\boldsymbol{A},\boldsymbol{B}$ are symmetric matrices, they can be diagonalized as $\boldsymbol{A} = \boldsymbol{U_A}\boldsymbol{\Sigma_A}\boldsymbol{U_A}^\top$, $\boldsymbol{B} = \boldsymbol{U_B}\boldsymbol{\Sigma_B}\boldsymbol{U_B}^\top$, where $\boldsymbol{\Sigma_A} = \text{diag}(\lambda_A^1,...,\lambda_A^{3p^2})$ and $\boldsymbol{\Sigma_B} = \text{diag}(\lambda_B^1,...,\lambda_B^M)$ are diagonal matrices. Since $\boldsymbol{A},\boldsymbol{B}$ are positive semi-definite matrices, their corresponding eigenvalues are non-negative, i.e., $\lambda_A^i\geq 0$ for $\forall i = 1,...,3p^2$ and $\lambda_B^j\geq 0$ for $\forall j = 1,...,M$. If at least one of the matrices $\boldsymbol{A},\boldsymbol{B}$ is positive definite, then $\lambda_A^i+\lambda_B^j>0$ holds true for $\forall i,j$ and $\sigma(\boldsymbol{A})\cap\sigma(-\boldsymbol{B}) = \emptyset$. Therefore, the Sylvester equation $\boldsymbol{AC}+\boldsymbol{CB}=\boldsymbol{E}$ has a unique solution. □