Image Super Resolution via Plug-and-Play ADMM and Its Solution Dependency on Denoising Performance

Thiti Premrudeepreechacharn (BME Senior Undergraduate) [1]

Abstract

Image Super Resolution (SR) problem is an inverse problem in computational imaging field that can be formulated and solved using the alternating direction method of multiplier (ADMM), a method that reformulate an unconstrained optimization problem into an constrained one. Then, there is a recently proposed method which replaces a step in ADMM with a denoiser, called Plug-and-Play ADMM (PnP), and this method is proven to converge under some conditions. However, it is unclear whether a good denoiser will produce a good SR solution or not. In this paper, we perform quantitative experiments for each denoiser on a set of 10 images throughout varying configurations of PnP.

1. Introduction

Image restoration, or image reconstruction, tasks are usually characterized by the transformation of the observed degraded image $\boldsymbol{y} \in R^n$, in order to find the true, uncorrupted, image $\boldsymbol{x} \in R^n$ shown as (1)

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{e}, \tag{1}$$

where $\boldsymbol{H} \in R^{n \times n}$ denotes the degradation matrix and $\boldsymbol{e} \in R^n$ denotes the additive noise vector, with the Additive White Gaussian Noise (AWGN) as the most popular one. As a subset of that, Image Super-Resolution (SR) is a technique, or a class of techniques, that is aimed to increase the resolution of an imaging system which can be formulated as a maximum-a-posteriori (MAP) estimation problem (Delon & Houdard, 2018). In addition, as one will see, this unconstrained optimization problem can also be reformulated as a constrained one, and this conversion is a technique called alternating direction method of multiplier (ADMM) which is the main character in the plug-and-play (PnP) framework we are focusing on (Boyd et al., 2011).

*Equal contribution [1]Department of Biomedical Engineering, Purdue University, West Lafayette IN, USA. Correspondence to: Stanley Chan, Ph.D. <stanchan@purdue.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

1.1. MAP and ADMM

The main goal of MAP estimation is to maximize the posterior probability as shown in (2), which the $p(\boldsymbol{y} \mid \boldsymbol{x})$ denotes the likelihood term, acting as the forward model, and the $p(\boldsymbol{x})$ denotes the prior distribution or probability distribution of the true image, acting as the regularization term

$$\begin{aligned} \hat{\boldsymbol{x}} &= \underset{\boldsymbol{x}}{\text{argmax}}\ p(\boldsymbol{x} \mid \boldsymbol{y}) \\ &= \underset{\boldsymbol{x}}{\text{argmin}}\ -log\, p(\boldsymbol{y} \mid \boldsymbol{x}) - log\, p(\boldsymbol{x}). \end{aligned} \tag{2}$$

From that, after applying the monotonic negative logarithm function, the problem could be simplified by assigning $f(\boldsymbol{x}) \overset{\text{def}}{=} -log\, p(\boldsymbol{y}|\boldsymbol{x})$ and $g(\boldsymbol{x}) \overset{\text{def}}{=} -(1/\lambda)\, log\, p(\boldsymbol{x})$ with λ as the confidence term, shown in (3)

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\text{argmin}}\ f(\boldsymbol{x}) + \lambda\, g(\boldsymbol{x}). \tag{3}$$

Then, ADMM converts this unconstrained optimization problem into a constrained one, shown in (4), thus splitting the variable which one could think of as splitting the problem into subproblems that, individually, can be more optimally solved

$$(\hat{\boldsymbol{x}}, \hat{\boldsymbol{v}}) = \underset{\boldsymbol{x}, \boldsymbol{v}}{\text{argmin}}\ f(\boldsymbol{x}) + \lambda g(\boldsymbol{v}), \text{subject to } \boldsymbol{x} = \boldsymbol{v}. \tag{4}$$

Moreover, when augmented Lagrangian method which adds the Lagrange multiplier variable (u) to the problem is applied to (4), shown as the function $\mathcal{L}$ in (5)

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{u}) = f(\boldsymbol{x}) + \lambda g(\boldsymbol{v}) + \boldsymbol{u}^T(\boldsymbol{x} - \boldsymbol{v}) + \frac{\rho}{2} \|\boldsymbol{x} - \boldsymbol{v}\|^2, \tag{5}$$

the solution to the problem can be found through solving subproblems to update the variables sequentially, addressed as (6), (7), and (8)

$$\boldsymbol{x}^{(k+1)} = \underset{\boldsymbol{x} \in R^n}{\text{argmin}}\ f(\boldsymbol{x}) + \frac{\rho}{2} \left\| \boldsymbol{x} - \tilde{\boldsymbol{x}}^{(k)} \right\|^2, \tag{6}$$

$$\boldsymbol{v}^{(k+1)} = \underset{\boldsymbol{v} \in R^n}{\text{argmin}}\ \lambda g(\boldsymbol{v}) + \frac{\rho}{2} \left\| \boldsymbol{v} - \tilde{\boldsymbol{v}}^{(k)} \right\|^2, \tag{7}$$

$$\bar{\boldsymbol{u}}^{(k+1)} = \bar{\boldsymbol{u}}^{(k)} + (\boldsymbol{x}^{(k+1)} - \boldsymbol{v}^{(k+1)}), \tag{8}$$

where $\bar{\boldsymbol{u}}^{(k+1)} \overset{\text{def}}{=} (1/\rho)\boldsymbol{u}^k$ is the scaled Lagrange Multiplier, tuned by ρ, $\tilde{\boldsymbol{x}}^{(k)} \overset{\text{def}}{=} \boldsymbol{v}^{(k)} - \bar{\boldsymbol{u}}^{(k)}$ and $\tilde{\boldsymbol{v}}^{(k)} \overset{\text{def}}{=} \boldsymbol{x}^{(k+1)} + \bar{\boldsymbol{u}}^{(k)}$ are temporary variables. In addition, it should be noted that there is an important assumption on f and g that can guarantee convergence of (6)-(8) to the solution of (3), which is that f and g are closed, proper, and convex.

1.2. Plug-and-Play ADMM

Using the structure of sequential update of ADMM, and its nature of splitting into subproblems, we can reformulate the role of the variables and its update to fit the problem we want to solve. Focusing on the image restoration problem, (6) could be seen as an inversion step, with the forward model $f(\boldsymbol{x})$ being involved, and (7) could be seen as a denoising step, with the prior $g(\boldsymbol{v})$ being involved, and (8) being just the update of the Lagrange multiplier. With that, further simplification and variable allocation could be applied without the loss of generality to achieve the intuition of the process. For example, defining $\sigma = \sqrt{\frac{\lambda}{\rho}}$, Chan et al. (Chan et al., 2016) showed that (7), which is considered as the denoising step since it involves the prior $g(\boldsymbol{v})$, can be rewritten as (9)

$$\boldsymbol{v}^{(k+1)} = \underset{\boldsymbol{v}\in R^n}{\operatorname{argmin}}\ g(\boldsymbol{v}) + \frac{1}{2\sigma^2}\left\|\boldsymbol{v} - \tilde{\boldsymbol{v}}^{(k)}\right\|^2. \tag{9}$$

This reformulation gave us the intuition that $\tilde{\boldsymbol{v}}^{(k)}$ is the noisy image, and σ is the denoising strength or the hypothesized noise level, which this hypothesized value could be understood as the prior. Additionally, it shows that (9) is trying to minimize the residue between $\tilde{\boldsymbol{v}}^{(k)}$ and the clean image $\boldsymbol{v}$, as interpreted by the algorithm, via the use of prior (g(v)). With (9) being controlled by both the prior and σ, Venkatakrishnan et al. (Venkatakrishnan et al., 2013), has proposed that (9) can be simplified further as (10)

$$\boldsymbol{v}^{(k+1)} = \mathcal{D}_\sigma(\tilde{\boldsymbol{v}}^{(k)}), \tag{10}$$

which $\mathcal{D}_\sigma$ denotes a proximal map or an off-the-shelf image denoisers. This usage of $\mathcal{D}_\sigma$ motivated the term 'Plug-and-Play' to be coined, since the user can plug in their own denoisers, and then the algorithm will play or solve your inverse problem via the ADMM framework. In addition, this heuristic nature begs the question if better denoisers consistently give us better results, whether in the general or a specific domain, or not. From that, there are modifications of the original PnP framework which assumes ρ to be fixed, with Chan et al. (Chan et al., 2016) proposing a continuation scheme by constantly updating value of ρ by $\rho_{k+1} = \gamma\rho_k$, called 'monotone update' rule which, note the k subscript of ρ_k, adds another layer to the overall PnP ADMM framework as follows:

$$\boldsymbol{x}^{(k+1)} = \underset{\boldsymbol{x}\in R^n}{\operatorname{argmin}}\ f(\boldsymbol{x}) + \frac{\rho_k}{2}\left\|\boldsymbol{x} - (\boldsymbol{v}^{(k)} - \boldsymbol{u}^{(k)})\right\|^2, \tag{11}$$

$$\boldsymbol{v}^{(k+1)} = \mathcal{D}_{\sigma_k}(\tilde{\boldsymbol{v}}^{(k)}), \tag{12}$$

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + (\boldsymbol{x}^{(k+1)} - \boldsymbol{v}^{(k+1)}), \tag{13}$$

$$\rho_{k+1} = \gamma_k\rho_k. \tag{14}$$

In addition, there is also an 'adaptive update' rule which update ρ in the same vein as that of the monotone update rule if value of the new residue

$$\Delta_{k+1} \overset{\text{def}}{=} \frac{1}{\sqrt{n}}(\epsilon_1 + \epsilon_2 + \epsilon_3), \tag{15}$$

where $\epsilon_1 \overset{\text{def}}{=} \left\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\right\|_2$, $\epsilon_2 \overset{\text{def}}{=} \left\|\boldsymbol{v}^{(k+1)} - \boldsymbol{v}^{(k)}\right\|_2$, and $\epsilon_1 \overset{\text{def}}{=} \left\|\boldsymbol{u}^{(k+1)} - \boldsymbol{u}^{(k)}\right\|_2$, is greater than or equal to that of a portion of the old residue, $\Delta_{k+1} \geq \eta\Delta k$, with $\eta \in [0, 1)$. In addition, (15) can also be used as a stop criteria, which Chan et al. (Chan et al., 2016) had determined that $\Delta_{k+1} \leq 10^{-3}$ is a good stop criteria, with 10^{-3} being the maximum value of this tolerance for stopping the algorithm, or say that the algorithm has converged in Algorithm 1. Then, assuming λ is fixed, the overall PnP ADMM procedure for the adaptive update rule is illustrated in Algorithm 1. Note that this adaptive update rule is inspired by the work by Goldstein et al. (Goldstein et al., 2014) which attempts to accelerate the ADMM problem via the introduction of the parameter η as an additional parameter that is the factor in consideration, whether to update or not, with the combined residual. Then, the convergence of the PnP ADMM scheme was analyzed by Chan et al. (Chan et al., 2016) which showed that for any bounded denoisers, the PnP ADMM with adaptive learning rule has a fixed point convergence, complementing the global convergence results presented in (Sreehari et al., 2016). Moreover, Ryu et al. (Ryu et al., 2019) presented another assumption other than assuming $2\mathcal{D}_\sigma - I$ is nonexpansive, a strong assumption that guarantees convergence of PnP ADMM, which assume this $\mathcal{D}_\sigma : R^n \to R^n$ for all $\boldsymbol{x}, \boldsymbol{y} \in R^n$ for some $\epsilon \geq 0$:

$$\|(\mathcal{D}_\sigma - I)(x) - (\mathcal{D}_\sigma - I)(y)\|^2 \leq \epsilon^2\|x - y\|^2, \tag{a}$$

and this is in addition to the data fidelity term, or $f(\boldsymbol{x})$ term in (3), being assumed as μ-strongly convex. These assumptions, both the $f(\boldsymbol{x})$ being μ-strongly convex and (a) all play a role showing that the PnP ADMM converges, in addition to leveraging the equivalence of PnP ADMM and the 'Plug-and-Play Douglas-Rachford Splitting' (PnP DRS) scheme. However, it must be noted that the strong convexity assumption is usually satisfied in image deblurring and image denoising applications, not the image super resolution and compressed sensing.

1.3. Related Works

After its initial introduction by Venkatakrishnan et al. (Venkatakrishnan et al., 2013) in 2013, the intuition of 'Plug-and-Play' has been implemented in combination with other schemes. For example, the Iterative Shrinkage/Thresholding

Algorithm 1 Plug-and-Play ADMM with Adaptive Update

Input: $\rho_0, \lambda, \eta < 1, \gamma > 1$.
while Not Converge **do**
  $\boldsymbol{x}^{(k+1)} = \text{argmin}_{\boldsymbol{x}} f(\boldsymbol{x}) + (\rho_k/2)\left\|\boldsymbol{x} - \tilde{\boldsymbol{x}}^{(k)}\right\|^2$,
where $\tilde{\boldsymbol{x}}^{(k)} = (\boldsymbol{v}^{(k)} - \boldsymbol{u}^{(k)})$
  $\boldsymbol{v}^{(k+1)} = \mathcal{D}_{\sigma_k}(\boldsymbol{x}^{(k+1)} + \boldsymbol{u}^{(k)})$, where $\sigma_k = \sqrt{\lambda/\rho_k}$
  $\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + (\boldsymbol{x}^{(k+1)} - \boldsymbol{v}^{(k+1)}$
  if $\Delta_{k+1} \geq \eta \Delta k$ **then**
    $\rho_{k+1} = \gamma \rho_k$
  else
    $\rho_{k+1} = \rho_k$
  end if
  $k = k + 1$
end while

Algorithm (ISTA), an alternative to ADMM on solving the inverse problem formulated as (3) which the large dimensionality of the imaging data and non-differentiability of the regularizer $(\boldsymbol{x})$ has called for a need of proximal algorithms, a component to both ISTA and ADMM, to assist in solving these problems. These algorithms heavily rely on the proximal operator, $prox_f(\boldsymbol{z}) = \text{argmin}_{\boldsymbol{x}\in X}(f(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2)$, which can be seen as (6) and (7) being reformulated as $prox_f(\tilde{\boldsymbol{x}}^{(k)})$ and $prox_f(\tilde{\boldsymbol{v}}^{(k)})$, respectively, which when it comes to PnP scheme, the proximal operator becomes the denoiser, as seen in a transition from (7) to (10). However, the differences between the PnP-ISTA and PnP-ADMM are on the inversion step which the proximal operator is still being used for PnP-ADMM as (6), in contrast to the PnP-ISTA which relies on the gradient of the data fidelity term, or $f(\boldsymbol{x})$ term in (3). The other difference lies in the Lagrange multiplier variable $\boldsymbol{u}$ update, as seen in Algorithm 2, proposed by Sun et al. (Sun et al., 2019). In addition, the convergence of PnP-ISTA is analyzed by another work (Xu et al., 2020) which uses MMSE denoiser with PnP-ISTA to prove that the overall scheme converges to a stationary point of some global cost function. It should be noted that if q_k variables in PnP-ISTA have their values adapted as $q_k = \frac{1}{2}(1 + \sqrt{1 + 4q_{k-1}^2})$, the algorithm, proposed by Beck and Teboulle (Beck & Teboulle, 2009) corresponds to the accelerated version of ISTA, called Fast ISTA, and one could consider it as PnP-FISTA which is being used in another work by Sun et al. (Sun et al., 2018) on Fourier ptychographic microscopy.

To consider another example of combining denoisers with another scheme, the approximate message passing (AMP) algorithms, which is a class of low-complexity, scalable algorithms for statistical estimation in problems such as compressed sensing (Rush & Venkataramanan, 2016). Stemming from that, Denoising-based AMP (D-AMP) is an extension of AMP by coupling with a denoiser, and it is shown

Algorithm 2 Plug-and-Play ISTA

Input: $\rho, \lambda, \gamma > 1$, and $q_{k\,k\in N}$.
while Not Converge **do**
  $\boldsymbol{x}^{(k+1)} = \boldsymbol{u}^{(k)} - \rho \nabla f(\boldsymbol{u}^{(k)})$
  $\boldsymbol{v}^{(k+1)} = \mathcal{D}_{\sigma_k}(\boldsymbol{x}^{(k+1)})$, where $\sigma_k = \sqrt{\lambda/\rho}$
  $\boldsymbol{u}^{(k+1)} = \boldsymbol{v}^{(k+1)} + ((q_{k-1} - 1)/q_k)(\boldsymbol{v}^{k+1} - \boldsymbol{v}^k)$
end while

by Metzler et al. (Metzler et al., 2014) that by plugging in a denoiser, it offers, or plays out, state-of-the-art compressed sensing recovery which is better than the use of AMP algorithms alone, in addition to being extremely robust to noise. However, it was noted that the framework heavily relies on the assumption of the residual signals following Gaussian distribution, in other words, the $\boldsymbol{A}$ in its specific forward model of $f(\boldsymbol{x}) = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|^2$. are strong assumptions that the behavior of D-AMP becomes ambiguous should the assumptions not be met. From that, a more recent robust alternative of AMP, called Vector AMP (VAMP) is proposed by Rangan et al. (Rangan et al., 2016), addresses the Gaussian assumption in that VAMP applies to arbitrarily conditioned matrices $\boldsymbol{A}$, and Fletcher et al. (Fletcher et al., 2018) had shown in a compressive image recovery task that VAMP only shows a miniscule amount of PSNR degradation due to ill-conditioned $\boldsymbol{A}$, when compared with that of AMP, coupled with the same denoisers. However, we are focusing only on Plug-and-Play nature of the ADMM algorithm.

1.4. Image Super-Resolution

From (3), which mentions the forward model as $f(\boldsymbol{x})$, the image super-resolution tasks, which consists of two operations: an anti-aliasing filter and a sub-sampling process, reformulate the data fidelity term $f(\boldsymbol{x})$ to be in the form

$$f(\boldsymbol{x}) = \|\boldsymbol{S}\boldsymbol{H}\boldsymbol{x} - \boldsymbol{y}\|^2, \tag{16}$$

where $\boldsymbol{H} \in R^{n\times n}$. is a binary matrix denoting the binary sampling matrix, in this case a K-fold downsampling, and $\boldsymbol{S} \in R^{m\times n}$ where $m < n$ is a convolution matrix representing the anti-aliasing filter. As noted in the user guide of MATLAB implementation of PnP for image restoration applications developed by Chan et al. (Chan et al., 2016), $\boldsymbol{H}$ is implemented using convolution $\boldsymbol{y} = \boldsymbol{h} * \boldsymbol{x}$, which $\boldsymbol{x}$ denotes the input, or the clean image in this case, and $\boldsymbol{h}$ denotes the blur kernel which could be implemented as either a bicubic blur kernel or a Gaussian blur kernel with specified standard deviation.

In addition, defining $\boldsymbol{G} = \boldsymbol{SH}$, (6) can be reformulated as

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in R^n}{\operatorname{argmin}} \ \|\boldsymbol{Gx} - \boldsymbol{y}\|^2 + \frac{\rho}{2}\|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|^2, \tag{17}$$

where we drop iteration number k and consider $\boldsymbol{x}^{(k+1)}$ as $\hat{\boldsymbol{x}}$ to analyze (17) in an optimization setting, which the solution being the pseudo-inverse

$$\hat{\boldsymbol{x}} = (\boldsymbol{G}^T\boldsymbol{G} + \rho\boldsymbol{I})^{-1}(\boldsymbol{G}^T\boldsymbol{y} + \rho\tilde{\boldsymbol{x}}). \tag{18}$$

As $\boldsymbol{G}^T\boldsymbol{G}$, or $\boldsymbol{H}^T\boldsymbol{S}^T\boldsymbol{SH}$ is neither diagonal nor diagonalizable by the Fourier transform, solving this f-subproblem becomes nontrivial. However, this problem has a closed-form solution when S is a standard K-fold downsampler and H is a circular convolution, using Sherman-Morrison-Woodbury identity, also known as the Woodbury matrix identity, to reformulate the pseudo-inverse solution as

$$\hat{\boldsymbol{x}} = \rho^-1\boldsymbol{b} - \rho^-1\boldsymbol{G}^T(\rho\boldsymbol{I} + \boldsymbol{GG}^T)^-1\boldsymbol{Gb}, \tag{19}$$

where $\boldsymbol{b} \overset{\text{def}}{=} \boldsymbol{G}^T\boldsymbol{y} + \rho\tilde{\boldsymbol{x}}$, following by the realization that $\boldsymbol{GG}^T$ gives us $\boldsymbol{SHH}^T\boldsymbol{S}^T$ which is a 'upsample-filter-downsample' sequence since $\boldsymbol{S}$ is a K-fold downsampling operator, $\boldsymbol{S}^T$ is a K-fold upsampling operator, and $\boldsymbol{HH}^T$ is a convolution between the blur kernel and its time-reversal. Consequently from this reformulation and realization of the sequence, one could use a concept of polyphase decomposition, initially introduced by the work of polyphase network analysis by Bellanger et al. (Bellanger et al., 1976), to decompose the z-transform representation of the blur matrix

$$\tilde{H}(z) = \sum_{k=0}^{K-1} z^{-k}\tilde{H}_k(z^K), \tag{20}$$

with $\tilde{H}_k(z^K)$ being the kth polyphase component of H(z). Then, as Vaidyanathan (Vaidyanathan, 1990) had shown the application of Noble identity on retrieving equivalent representation of the polyphase decomposition, the Noble identity can be applied on the $\boldsymbol{GG}^T$ operation which made the overall sequence of $\boldsymbol{SHH}^T\boldsymbol{S}^T$ to be simplified in a form of a finite impulse response (FIR) filter $\tilde{H}_0(z)$, the 0th polyphase component of the filter $\boldsymbol{HH}^T$. Consequently, by downsampling the convolved filter $\tilde{\boldsymbol{H}} = \boldsymbol{HH}^T$, the 0th polyphase component is implemented, and the FIR nature of the sequence makes the pseudo-inverse solution (19) to be rewritten as a closed-form solution with the help of Fourier transform

$$\boldsymbol{x} = \rho^-1\boldsymbol{b} - \rho^-1\boldsymbol{G}^T(\mathcal{F}^{-1}\frac{\mathcal{F}(\boldsymbol{Gb})}{|\mathcal{F}(\tilde{h}_0)|^2 + \rho}), \tag{21}$$

where $\boldsymbol{b} = \boldsymbol{G}^T\boldsymbol{y} + \rho\tilde{\boldsymbol{x}}$. From that, Chan et al. (Chan et al., 2016) has quantitatively shown that this closed-form method has significantly reduced the runtime of solving this inversion step, compared to the conjugate gradient method (Brifman et al., 2016), in which the runtime increases linearly with increasing iterations on the log-log plot.

1.5. Contributions

With the recent developments of the Plug-and-Play ADMM methods (PnP), both in terms of comparable image super resolution performance, if not superior, compared to that of other methods such as the transformed self-exemplar method (TSE) by Huang et al. (Huang et al., 2015) or the Gaussian process regression method (GPR) by He and Siu (He & Siu, 2011), and in terms of convergence analyzed by Chan et al. (Chan et al., 2016), it can be said that PnP has earned its place as a state-of-the-art methods for solving the image super resolution (SR) problem. However, there is still a gap in the understanding of the optimal choice of denoiser that is being plugged into the algorithm, which there are postulates based on the intuition that superior denoiser implies a superior PnP performance.

From that, we attempt to address this gap by conducting quantitative experiment of PnP for solving SR problem using 10 of the well-known Set12 dataset (Zhang et al., 2017), by first defining the term 'superior denoiser', which in this case is going to be the denoiser that has superior denoising performance measured with PSNR, compared to the others at all noise levels. Then, by pinpointing the superior denoiser, we plug them into the PnP algorithm, and measure the PSNR of the output using the original image as the reference. By averaging the PSNR over all the denoisers, we hypothesize that, using the same parameters, the denoiser that perform denoising well at the specified noise level will also solve the SR problem well, in a positively correlated way. In other words, if that specified denoiser performs denoising well at noise level of 10/255, but poorly at noise level of 50/255, we hypothesize that this denoiser will solve the SR problem via PnP well at noise level of 10/255, but poorly at noise level of 50/255.

2. Methods

2.1. Dataset

Prior to assessing the performance of the SR model, we used 10 standard test images in .png from the Set12 dataset, a well known 12 grayscale image dataset initially introduced by Zhang et al. (Zhang et al., 2017), as a benchmark for image denoising methods. The images are shown in 1 as the first row images have a size of 256x256 and the second row images have a size of 512x512. Then, after implementing blur kernel, downsampling, and adding Gaussian noise to the original image, we also use the same set of images to test the SR performance with the scaling factor of K = 2. The reason behind the usage of same images for SR task is that we would like to observe the correlation between the denoising performance, implying the potential of the denoiser, and the SR performance, which the correlation might be ambiguous should the input image be different.

Figure 1. 10 testing images for the experiment

2.2. Experimental Setup

2.2.1. Denoising Task

With the dataset in hand, we first apply AWGN with σ = 10/255, 25/255, 50/255, 70/255, 100/255, 130/255, 160/255, and 190/255 as 8 separate cases. Then, we normalize the noisy images to make a preferable data range of [0,1] for the denoisers, which in this case we have chosen to implement the denoising task using the Recursive Filter (RF) (Gastal & Oliveira, 2011), Non-local Means (NLM) (Buades et al., 2011), Block-matching and 3D Filtering (BM3D) (Dabov et al., 2006), Total Variation (TV) (Chan et al., 2011), Deep Convolutional Neural Network (DnCNN) (Zhang et al., 2017), and Fast and Flexible Denoising Convolutional Neural Network (FFDNet) (Zhang et al., 2018) denoisers. Specifically, we have benefitted from the MATLAB implementation of PnP for image restoration applications developed by Chan et al.(Chan et al., 2016) by using the wrapper function that links directly to the function activating the denoiser RF, NLM, BM3D, TV which we kept the constants of each and every case the same, and all denoisers are available for download as an open-source software. DnCNN denoiser is, on the other hand, available as a sequence of calling MATLAB readily available functions, `net = denoisingNetwork('DnCNN')` for retrieving a pretrained image denoising convolutional neural network which then becomes an input parameter for the function `denoised = denoiseImage(noisy,net)`, estimating the denoised image, `denoised`, from noisy image, `noisy`, using the pre trained DnCNN specified by `net`. However, it must be noted that these two functions require a Deep Learning Toolbox (The MathWorks, 2018). Then, using the MATLAB implementation of FFDNet developed by Zhang et al. (Zhang et al., 2018) and inspired by the practicality offered by the aforementioned wrapper function, we have written the wrapper function for the FFDNet, which is implemented by editing the script Demo_AWGN_Gray.m into our customized script FFDNet.m which loads the pre trained FFDNet model for grayscale images, FFDNet_gray.mat, and perform denoising on the noisy image input by evaluating the network model via function vl_simplenn.m. Note that this vl_simplenn.m function is from an external open-source MATLAB toolbox for implementing CNNs called MatConvNet (Vedaldi & Lenc, 2015).

Consequently after performing denoising task, we quantify the performance using the MATLAB function, `psnr(denoised_img, ground_truth)` which calculates, in dB, the peak signal-to-noise ratio (PSNR) of the denoised image, `denoised_img`, with the image `ground_truth` as the reference. The PSNR values of all images are then averaged to look at the overall performance. In addition, this method is also applied on the SR task, comparing the PnP output with the original grayscale image.

2.2.2. Image Super-Resolution

After the denoising tests are done, the next phase of the experiments is to put the images of Set12 dataset to go through the PnP algorithm for image SR task at a factor of $K = 2$, which is using the lower resolution image as input into the algorithm and having a high resolution image as an output, with a hope that the output and the ground truth are similar. Having known that the BM3D is a state-of-the-art denoiser Chan et al. (Chan et al., 2016) used to analyze SR performance of PnP ADMM algorithm, which surpasses other methods being discussed, we plan to compare the SR performance using BM3D with the case of using DnCNN, in addition to the case of a faster alternative, FFDNet, in order to prove our hypothesis. In addition, even though the behavior of adaptive update rule for the PnP ADMM scheme shown in Algorithm 1 is proven to be more robust than that of the monotone update and the original method proposed by Venkatakrishnan et al. (Venkatakrishnan et al., 2013), we have chosen to perform SR task on those two cases as well, in order to probe into the robustness of the algorithm on a particular denoiser, in addition to determining the final product if the choice of denoiser uniformly affects SR results. Note that in addition to measuring the PSNR and averaging the PSNR over all the images, one should keep in mind that the upscaling procedure of the image in this case utilizes only the bicubic interpolation method, or the function `imresize(img,K)` in MATLAB, where `img` corresponds to the image and `K` corresponds to the factor of magnification, 2 in this case.

Moreover, the influence of ρ_0 impinged upon SR performance is not to be overlooked, which is why we have set up another degree of freedom on varying the value of initial ρ in order to both verify the result from Chan et al. (Chan et al., 2016) that ρ_0 = 1 is the best alternative, and to look at the performance of each denoiser iteration-wise, if it is robust or not, with a further discussion in Appendix 5.1. From that, we have chosen the values $\rho_0 = 10, 1, 10^{-2}, 10^{-5}$ which

are in the range of typical ρ_0 that perform well. However, the ρ_0 value of 10 was chosen to test the presumably convex nature of this parameter, in other words, if the SR performance for $\rho_0 = 1$ is truly the best, increasing ρ_0 to 10 should degrade the performance, same applies to $\rho_0 = 10^{-2}$.

Then, after the verification of the best ρ_0 being a value of 1, we perform the SR task on original PnP-ADMM, monotone update rule, and the adaptive update rule as the rationale are discussed above. In addition, it should be noted that the maximum number of iterations are decreased to 15 for both monotone and adaptive update rule, in contrast to the case of original PnP which has 20 iterations, because the monotone update rule applied to the denoisers TV and RF returned only 17 iterations of PnP outputs which could be explained by the residual (15) having a value greater than the tolerance set at 10^{-4} which is greater than the minimal acceptable tolerance level of 10^{-3} as discussed in Chan et al. (Chan et al., 2016) regarding the stopping criteria. Also, as we are ultimately looking at the SR performance of each denoiser with respect to the noise levels, 3 sets of experimental configurations are studied which each has varying noise levels, and are shown in Table 1, where λ_{BM3D} corresponds to the λ parameter for the BM3D denoiser, and vice versa for other respective denoisers.

Table 1. Configuration, Parameters, and Variables for SR task K=2; Mono = Mototone Update Rule; Adapt = Adaptive Update Rule; Iter = Maximum Iterations; **Params** = Parameters.

Config	Description
Original	$\boldsymbol{H}$ = 9×9 Gaussian of std 1, γ = 1, Iter = 20
Mono	$\boldsymbol{H}$ = 9×9 Gaussian of std 1, γ = 1.2, Iter = 15
Adapt	$\boldsymbol{H}$ = 9×9 Gaussian of std 1, γ = 1.2, Iter = 15, η = 0.75
Params	Noise = [5,10,25,50,70]/255, $\rho_0 = 1$, $\lambda_{RF} = 0.0002, \lambda_{NLM} = 0.001$, $\lambda_{BM3D} = 0.001, \lambda_{TV} = 0.01$, $\lambda_{DnCNN} = 0.01, \lambda_{FFDNet} = 0.01$

3. Results

3.1. Denoising Task

Depicted in Table 2, one could see that the DnCNN denoiser performed best, averaged through all noise levels, with NLM coming in at second place, FFDNet coming in close as a third place, BM3D coming in at the fourth place, TV coming in at the fifth place, and RF coming in at the sixth place.

Then, showing the best denoiser at a particular noise level in bold, a trend of FFDNet performing better than other denoisers at high noise levels and its poor performance at low noise levels can be explained by the parameter being input into the denoiser, σ, or the noise level that the algorithm will estimate as the true noise level of the noisy image, which we have chosen a value of 0.1 or approximately 25/255, coming from $\sigma = \sqrt{0.01/1}$, meaning that the algorithm will estimate the noise level to be 25/255. However, the high noise level map of FFDNet outputs an inferior performance to DnCNN overall, which performs the best in this case. In addition, one should also note the superior performance of BM3D at low noise levels. For further visualization of the denoising performance, Figure 2 gives a pictorial illustration of the denoising performance of the denoisers, which it is easier to see the poor performance of FFDNet at low noise levels, the superior performance of BM3D at low noise levels, the uniformly superior performance of DnCNN, and the superior performance of FFDNet at high noise levels which is comparable to that of the DnCNN.

Table 2. Denoising Performance via PSNR (dB) at Various Noise Levels

Noise (σ)	RF	NLM	BM3D	TV	DnCNN	FFDNet
5	31.75	31.23	**31.94**	31.85	31.58	28.65
10	26.91	29.15	**29.63**	28.22	29.20	27.34
25	19.86	21.26	20.66	20.61	**21.68**	21.50
50	16.59	17.48	16.92	17.10	18.03	**18.08**
70	15.46	16.21	15.71	15.90	16.78	**16.88**
100	14.51	15.17	14.71	14.91	15.76	**15.86**
130	14.00	14.59	14.18	14.36	15.14	**15.20**
160	13.71	14.28	13.88	14.06	**14.83**	**14.83**
190	13.48	14.04	13.64	13.82	14.62	**14.63**
Average	18.47	19.27	19.03	18.98	**19.74**	19.22

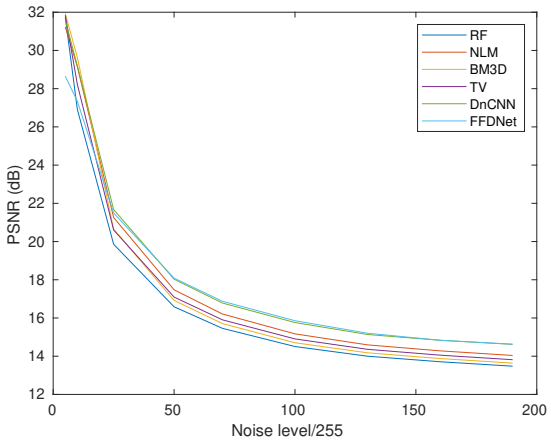

Figure 2. Denoising Performance at Various Noise Levels

3.2. Image Super-Resolution

After the validation of the best ρ_0 parameter value being the value of 1, discussed in 5.1, one can refer to Table 3 which depicts a similar trend of Image Super Resolution

(SR) Performance to that of the denoising performance in Table 2. In addition, for visual comparison on the SR task results, outputs of the PnP on adaptive update configuration is shown as Figure 3 which, interestingly, RF and DnCNN performs SR task even worse than using merely bicubic reconstruction for upscaling the downscaled and blurred image, as seen in the caption of Figure 3.

Table 3. Image Super Resolution ($K = 2$) Results at Various Noise Levels, all done using $\rho_0 = 1$; **Avg** = Average

	Denoisers					
Noise (σ)	RF	NLM	BM3D	TV	DnCNN	FFDNet
Original PnP Scheme; $\gamma = 1$; Maximum Iterations = 20						
5	25.10	24.93	**25.31**	25.28	24.72	22.91
10	24.85	24.99	**25.36**	25.03	22.74	22.74
25	18.72	**21.17**	21.15	20.80	17.77	20.10
50	14.95	17.50	17.17	17.52	14.88	**17.57**
70	13.80	16.15	15.65	16.34	13.83	**16.58**
Avg	19.48	20.95	20.93	**20.99**	18.78	19.98
Monotone Update; $\gamma = 1.2$; Maximum Iterations = 15						
5	24.91	24.52	24.97	**25.15**	24.80	22.55
10	24.76	24.56	**25.02**	24.60	23.64	22.37
25	19.50	**20.84**	20.76	20.27	19.20	19.93
50	16.27	17.41	17.16	17.18	16.24	**17.57**
70	15.16	16.21	15.89	16.07	15.17	**16.68**
Avg	20.12	20.71	**20.76**	20.65	19.81	19.82
Adaptive Update; ; $\gamma = 1.2$; $\eta = 0.75$; Maximum Iterations = 15						
5	24.95	24.61	25.08	**25.32**	24.86	22.55
10	24.80	24.70	**25.15**	24.81	23.58	22.37
25	19.44	**21.00**	20.94	20.49	19.10	19.91
50	16.18	17.48	17.19	17.32	16.14	**17.54**
70	15.06	16.23	15.86	16.20	15.06	**16.60**
Avg	20.09	20.80	**20.84**	20.83	19.75	19.80

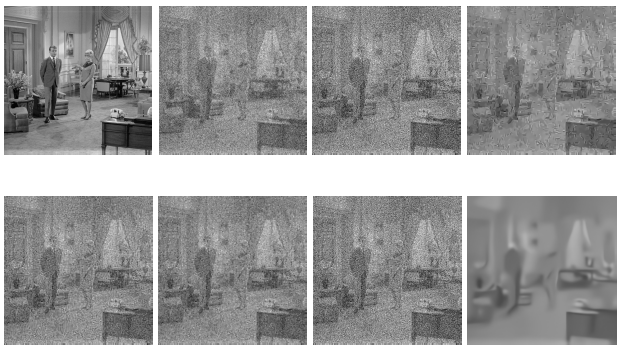

Figure 3. Image SR Results at $\sigma = 50$. [Top](from left to right). Ground truth; Bicubic reconstruction (17.37 dB); RF (16.94 dB); NLM (18.39 dB). [Bottom](from left to right). BM3D (18.12 dB); TV (18.16 dB); DnCNN (16.88 dB); FFDNet (18.25 dB)

The uniformity and similarity of the SR performance throughout all configurations can be visualized using Figure 4 which the original scheme depicted in blue color can be seen as having a superior performance at noise level of 5, 10, and 25, compared to that of the monotone update configuration depicted in red, and that of the adaptive update configuration depicted in black. However, both the monotone and adaptive update have a superior performance at higher noise levels of 50 and 70, compared to that of the original PnP scheme. Also, note that these gaps between configurations are not as significant as the gaps between the choice of denoisers, which one can observe via the example of a comparison between BM3D and FFDNet shown in Figure 5.

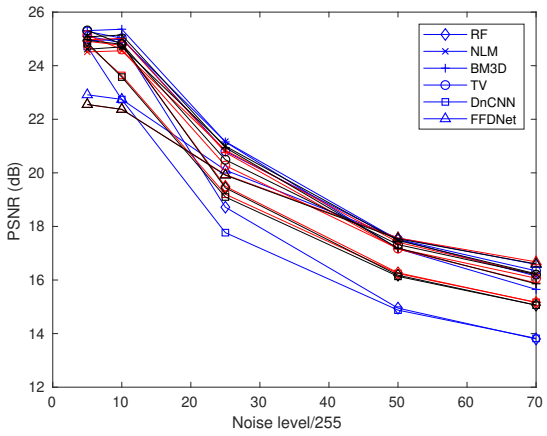

Figure 4. SR Performance at Various Noise Levels, with blue curve depicting the original scheme, red curve depicting monotone update configuration, and black curve depicting adaptive update scheme

We believe that the gap between method is due to the estimation of noise standard deviation, which is governed by the term $\sigma = \sqrt{\lambda/\rho}$, and this σ is integrated into the denoisers to estimate and attempt to denoise, an integral step depicted by (10). This estimation is updated in an iterative manner in both monotone and adaptive update configuration, using the term γ which explains the SR performance at higher noises being better than that of the original scheme. For example, BM3D has the input of sigma into their algorithm, which the denoiser has a fixed λ value of 0.001, inferring that at ρ value of 1, the estimated σ, or noise level, is approximately 8. Then, as FFDNet has a fixed λ value of 0.01, and following the same method, the estimated noise level becomes 25.5.

From that, Figure 5 helps visualize the performance of BM3D which performs well at lower noise levels, and confirming the observation on the effects of σ estimation which varies between configurations. FFDNet, on the other hand, has a significant gap of SR performance at lower noise levels when compared with that of BM3D, and better performance at higher noise levels, all that in addition to the confirmation of the aforementioned observation.

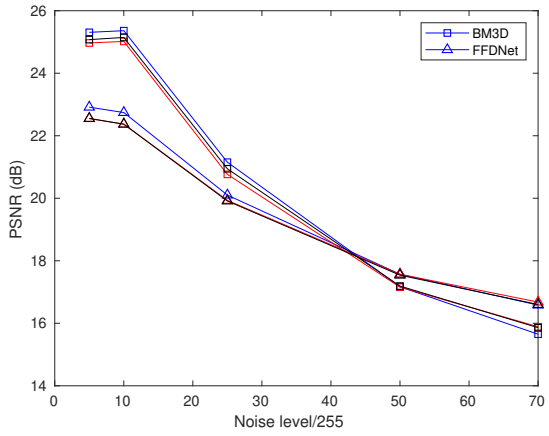

Figure 5. SR Performance of BM3D versus FFDNet at Various Noise Levels and Configurations

In addition, the performance of TV denoiser which had the same λ value as FFDNet, 0.01, on SR task can be seen on Figure 6 which shows a significant gap in SR performance at lower noise levels when compared with FFDNet. Also, the TV denoiser performed SR task better than BM3D at higher noise levels, suggesting that λ values truly plays a role in outputting better SR performance at a particular noise level. Nonetheless, there are some discrepancies in the choice of denoiser, as a significant gap between TV and FFDNet is present in denoising performance at noise level higher than 50, but the gap is almost nonexistent for the SR performance, considering TV and FFDNet.

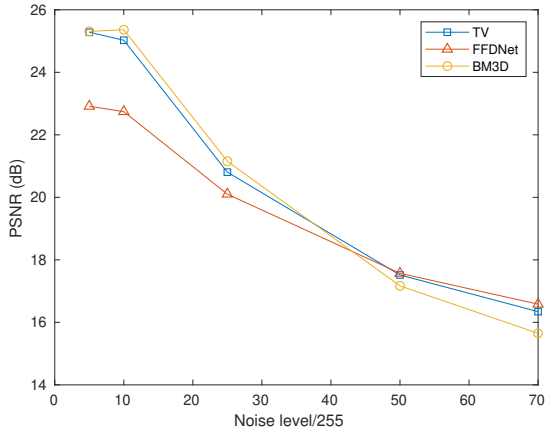

Figure 6. Comparison of SR Performance for FFDNet, BM3D, and TV at Various Noise Levels using Original Scheme

However, these gaps and similarity in trend are not present in the DnCNN denoiser, which performs best, on average, with high denoising potential at higher noise levels, but its poor performance on SR significantly contradicts our findings on the FFDNet. We have postulated that this contradiction for the DnCNN case stems from either its function in MATLAB which is a built-in function lacking the tunable parameter that will be tuned in the monotone and adaptive configuration, σ, which , or its pre-trained nature such that it might not perform very well on the output of the inversion step, which it would be essential to test on other inverse problems related to SR problem such as the deblurring problem which is the SR problem without the downscaling and upscaling module. In addition, the DnCNN might be an unbounded denoiser, one which violates the condition leading to convergence, as proven by Chan et al. (Chan et al., 2016).

Then, both Figure 2 and Figure 4 suggests a positive correlation in the choice of denoiser and the SR performance, but that DnCNN makes a contradictory case with possible cause of discrepancy above. This is with a note that choice of configuration, ρ_0, and λ, which infers on the parameter of denoiser σ, playing an important role in bringing about this correlation, but not as important as the intrinsic property of the denoiser itself, as one can take the TV denoiser compared to FFDNet example. Moreover, one should keep in mind the tunable parameters and the nature of the denoiser, as mentioned in the DnCNN case.

4. Conclusion

We presented a quantitative experimental result for Plug-and-Play ADMM used to solve image super resolution problem. We showed that the denoisers being plugged into the ADMM algorithm has a positively correlated performance between the denoising performance and image super resolution solving performance, but not a perfectly positive correlation and there are outliers present. The contradictory results, positively correlated performance on FFDNet and vague results on DnCNN, lead to a conclusion that a uniformly superior denoiser does not strictly imply a stronger candidate of a denoiser to be plugged into the ADMM algorithm. In addition, we experimentally show that at an optimum ρ_0 value of 1, the behavior across the experimental configurations are consistent.

5. Appendix

5.1. Appendix A: Effects of Varying ρ_0 Parameter

As mentioned in Section 2.2.2 regarding the choice of ρ_0 in consideration, $\rho_0 = [10, 1, 10^{-2}, 10^{-5}]$, we attempt to confirm the literature results on that $\rho_0 = 1$ is the best alternative out of the four via the use of the original PnP scheme configuration with the same parameters depicted in Table 1. Moreover, we are also looking at the performance iterative-wise, in order to identify if the performance for the particular denoiser at a noise level is robust or not.

From that, the only variable at play is the ρ_0 value, and by plugging different and the same test images for the other experiments, with different noise levels, we have calculated the PSNR, and consequently average them over all the test images and stored for each iteration, which the results are shown as Figure 7, 8, and 9. Note that these figures vary in noise levels in order to look at the SR performance, which is assumed to be varied throughout different noise levels.

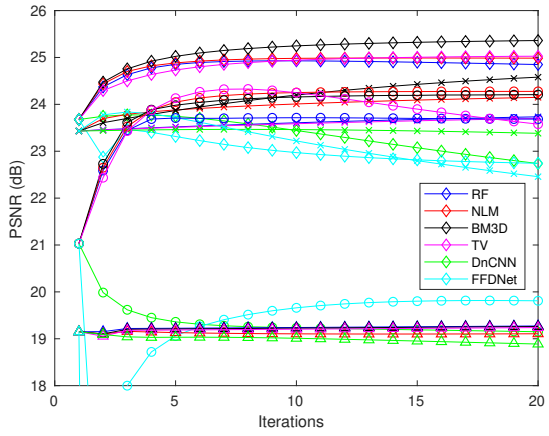

Figure 7. SR Performance at Each Iterations for Varying ρ_0 Values at $\sigma = 10$; Diamond: $\rho_0 = 1$; Cross (X):$\rho_0 = 10$; Circle (o):$\rho_0 = 10^{-2}$; Triangle: $\rho_0 = 10^{-5}$

By focusing on the ρ_0 values first, Figure 7 shows that at $\rho_0 = 1$, for every denoisers, except FFDNet and DnCNN, the SR performance turns out to be superior to other ρ_0. Note that $\rho_0 = 10$ and $\rho_0 = 10^{-5}$ had a lesser degree of fluctuation than that of $\rho_0 = 1$ and $\rho_0 = 10^{-2}$ which had the most fluctuation, despite a significant gap between itself and the $\rho_0 = 1$ case. Also, the constant degradation of DnCNN and FFDNet at $\rho_0 = 1$ and $\rho_0 = 10$, respectively, are not to be overlooked, since it might explains the aberrant behavior of DnCNN for playing in the ADMM algorithm.

Then, this Figure 8 shows the same ultimate result of $\rho_0 = 1$ being the best alternative, with an interesting finding that FFDNet at $\rho_0 = 1$ performs superior to other denoisers, and that TV denoiser at $\rho_0 = 10^{-2}$ has a significant amount of increase in PSNR over just 10 iterations. Also, it can be seen that every denoiser, except DnCNN and RF at $\rho_0 = 1$, has a robust performance after iteration number 15.

Figure 9 also suggests the same trend as that of Figure 8, in addition to the TV, FFDNet, and NLM being the ideal denoiser for higher noise levels at $\rho_0 = 1$ and 10^{-2} which was not mentioned. However, note that the SR performance of FFDNet at $\rho_0 = 10^{-5}$, for all noise levels, deteriorates after iteration number 2 (not shown), and plummets to a negative PSNR value which made $\rho_0 = 10^{-5}$ an infeasible value for FFDNet case.

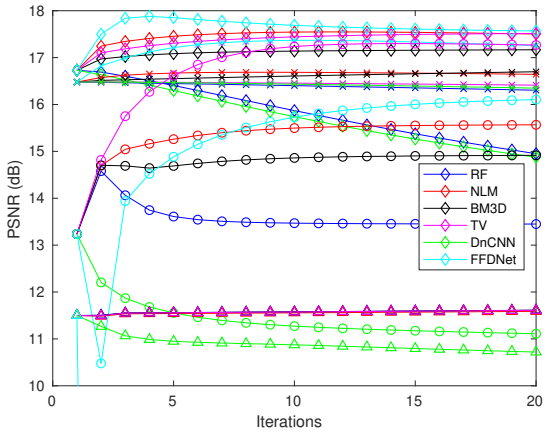

Figure 8. SR Performance at Each Iterations for Varying ρ_0 Values at $\sigma = 50$; Diamond: $\rho_0 = 1$; Cross (X):$\rho_0 = 10$; Circle (o):$\rho_0 = 10^{-2}$; Triangle: $\rho_0 = 10^{-5}$

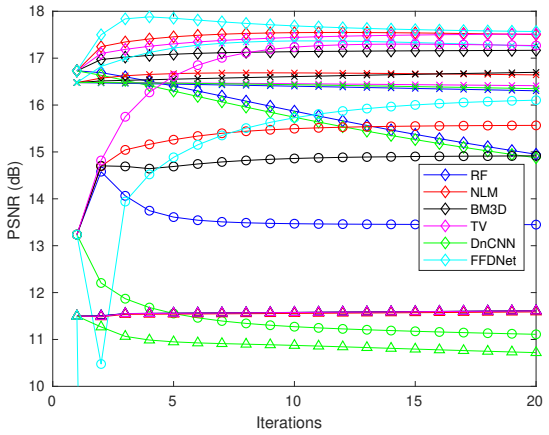

Figure 9. SR Performance at Each Iterations for Varying ρ_0 Values at $\sigma = 70$; Diamond: $\rho_0 = 1$; Cross (X):$\rho_0 = 10$; Circle (o):$\rho_0 = 10^{-2}$; Triangle: $\rho_0 = 10^{-5}$

References

Beck, A. and Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

Bellanger, M., Bonnerot, G., and Coudreuse, M. Digital filtering by polyphase network:application to sample-rate alteration and filter banks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(2):109–114, 1976. doi: 10.1109/TASSP.1976.1162788.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, January 2011. ISSN

1935-8237. doi: 10.1561/2200000016. URL https://doi.org/10.1561/2200000016.

Brifman, A., Romano, Y., and Elad, M. Turning a denoiser into a super-resolver using plug and play priors. In *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 1404–1408, 2016. doi: 10.1109/ICIP.2016.7532589.

Buades, A., Coll, B., and Morel, J.-M. Non-local means denoising. *Image Processing On Line*, 1:208–212, 2011.

Chan, S. H., Khoshabeh, R., Gibson, K. B., Gill, P. E., and Nguyen, T. Q. An augmented lagrangian method for total variation video restoration. *IEEE Transactions on Image Processing*, 20(11):3097–3111, 2011.

Chan, S. H., Wang, X., and Elgendy, O. A. Plug-and-play admm for image restoration: Fixed point convergence and applications, 2016.

Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. Image denoising with block-matching and 3d filtering. In *Image Processing: Algorithms and Systems, Neural Networks, and Machine Learning*, volume 6064, pp. 606414. International Society for Optics and Photonics, 2006.

Delon, J. and Houdard, A. Gaussian priors for image denoising. In *Denoising of Photographic Images and Video*, pp. 125–149. Springer, 2018.

Fletcher, A. K., Rangan, S., Sarkar, S., and Schniter, P. Plug-in estimation in high-dimensional linear inverse problems: A rigorous analysis. *CoRR*, abs/1806.10466, 2018. URL http://arxiv.org/abs/1806.10466.

Gastal, E. S. and Oliveira, M. M. Domain transform for edge-aware image and video processing. In *ACM SIGGRAPH 2011 papers*, pp. 1–12. 2011.

Goldstein, T., O'Donoghue, B., Setzer, S., and Baraniuk, R. Fast alternating direction optimization methods. *SIAM Journal on Imaging Sciences*, 7(3):1588–1623, 2014.

He, H. and Siu, W.-C. Single image super-resolution using gaussian process regression. In *CVPR 2011*, pp. 449–456. IEEE, 2011.

Huang, J.-B., Singh, A., and Ahuja, N. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5197–5206, 2015.

Metzler, C. A., Maleki, A., and Baraniuk, R. G. From denoising to compressed sensing. *CoRR*, abs/1406.4175, 2014. URL http://arxiv.org/abs/1406.4175.

Rangan, S., Schniter, P., and Fletcher, A. K. Vector approximate message passing. *CoRR*, abs/1610.03082, 2016. URL http://arxiv.org/abs/1610.03082.

Rush, C. and Venkataramanan, R. Finite sample analysis of approximate message passing. *CoRR*, abs/1606.01800, 2016. URL http://arxiv.org/abs/1606.01800.

Ryu, E. K., Liu, J., Wang, S., Chen, X., Wang, Z., and Yin, W. Plug-and-play methods provably converge with properly trained denoisers, 2019.

Sreehari, S., Venkatakrishnan, S. V., Wohlberg, B., Buzzard, G. T., Drummy, L. F., Simmons, J. P., and Bouman, C. A. Plug-and-play priors for bright field electron tomography and sparse interpolation. *IEEE Transactions on Computational Imaging*, pp. 1–1, 2016. ISSN 2334-0118. doi: 10.1109/tci.2016.2599778. URL http://dx.doi.org/10.1109/TCI.2016.2599778.

Sun, Y., Xu, S., Li, Y., Tian, L., Wohlberg, B., and Kamilov, U. S. Regularized fourier ptychography using an online plug-and-play algorithm. *CoRR*, abs/1811.00120, 2018. URL http://arxiv.org/abs/1811.00120.

Sun, Y., Wohlberg, B., and Kamilov, U. S. An online plug-and-play algorithm for regularized image reconstruction. *IEEE Transactions on Computational Imaging*, 5(3):395–408, 2019.

The MathWorks, I. *Deep Learning Toolbox*. Natick, Massachusetts, United State, 2018. URL https://www.mathworks.com/help/deeplearning/index.html.

Vaidyanathan, P. P. Multirate digital filters, filter banks, polyphase networks, and applications: a tutorial. *Proceedings of the IEEE*, 78(1):56–93, 1990.

Vedaldi, A. and Lenc, K. Matconvnet – convolutional neural networks for matlab. In *Proceeding of the ACM Int. Conf. on Multimedia*, 2015.

Venkatakrishnan, S. V., Bouman, C. A., and Wohlberg, B. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pp. 945–948. IEEE, 2013.

Xu, X., Sun, Y., Liu, J., Wohlberg, B., and Kamilov, U. S. Provable convergence of plug-and-play priors with mmse denoisers. *IEEE Signal Processing Letters*, 27:1280–1284, 2020.

Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.

Zhang, K., Zuo, W., and Zhang, L. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018. doi: 10.1109/TIP.2018.2839891.