Provably Convergent Algorithms for Denoiser-Driven Image Regularization

A Thesis
Submitted for the Degree of
Doctor of Philosophy
in the Faculty of Engineering

by

Pravin Nair

Department of Electrical Engineering
INDIAN INSTITUTE OF SCIENCE
Bengaluru – 560 012
2022

Abstract

Some fundamental reconstruction tasks in image processing can be posed as an inverse problem where we are required to invert a given forward model. For example, in deblurring and superresolution, the ground-truth image needs to be estimated from blurred and low-resolution images, whereas in CT and MR imaging, a high-resolution image must be reconstructed from a few linear measurements. Such inverse problems are invariably ill-posed—they exhibit non-unique solutions and the process of direct inversion is unstable. Some form of image model (or prior) on the ground truth is required to regularize the inversion process. For example, a classical solution involves minimizing $f + g$, where the loss term f is derived from the forward model and the regularizer g is used to constrain the search space. The challenge is to come up with a formula for g that can yield good image reconstructions. This has been the center of research activity in image reconstruction for the last few decades.

"Regularization using denoising" is a recent breakthrough in which a powerful denoiser is used for regularization purposes, instead of having to specify some hand-crafted g (but the loss f is still used). This has been empirically shown to yield significantly better results than staple $f + g$ minimization. In fact, the results are generally comparable and often superior to state-of-the-art deep learning methods. In this thesis, we consider two such popular models for image regularization—Plug-and-Play (PnP) and Regularization by Denoising (RED). In particular, we focus on the convergence aspect of these iterative algorithms which is not well understood even for simple denoisers. This is important since the lack of convergence guarantee can result in spurious reconstructions in imaging applications. The contributions of the thesis in this regard are as follows.

PnP with linear denoisers: We show that for a class of non-symmetric linear denoisers that includes kernel denoisers such as nonlocal means, one can associate a convex regularizer g with the denoiser. More precisely, we show that any such linear denoiser can be expressed as the proximal operator of a convex function, provided we work with a non-standard inner product (instead of the Euclidean inner product). In particular, the regularizer is quadratic, but unlike classical quadratic regularizers, the quadratic form is derived from the observed data. A direct implication of this observation is that (a simple variant of) the PnP algorithm based on this linear denoiser amounts to solving an optimization problem of the form $f + g$, though it was not originally conceived this way. Consequently, if f is convex, both objective and iterate convergence are guaranteed for the PnP algorithm. Apart from the convergence

guarantee, we go on to show that this observation has algorithmic value as well. For example, in the case of linear inverse problems such as superresolution, deblurring and inpainting (where f is quadratic), we can reduce the problem of minimizing $f + g$ to a linear system. In particular, we show how using Krylov solvers we can solve this system efficiently in just few iterations. Surprisingly, the reconstructions are found to be comparable with state-of-the-art deep learning methods. To the best of our knowledge, the possibility of achieving near state-of-the-art image reconstructions using a linear solver has not been demonstrated before.

PnP and RED with learning-based denoisers: In general, state-of-the-art PnP and RED algorithms rely on trained CNN denoisers such as DnCNN. Unlike linear denoisers, it is difficult to place PnP and RED algorithms within an optimization framework in the case of CNN denoisers. Nonetheless, we can still try to understand the convergence of the sequence of iterates generated by these algorithms. For convex loss f, we show that this question can be resolved using the theory of monotone operators — the denoiser being averaged (a subclass of nonexpansive operators) is sufficient for iterate convergence of PnP and RED. Using numerical examples, we show that existing CNN denoisers are not nonexpansive and can cause PnP and RED algorithms to diverge. Can we train denoisers that are provably nonexpansive? Unfortunately, this is computationally challenging—simply checking nonexpansivity of a CNN is known to be intractable. As a result, existing algorithms for training nonexpansive CNNs either cannot guarantee nonexpansivity or are computation intensive. We show that this problem can be solved by moving away from CNN denoisers to unfolded deep denoisers. In particular, we are able to construct unfolded networks that are efficiently trainable and come with convergence guarantees for PnP and RED algorithms, and whose regularization capacity can be matched with CNN denoisers. Presumably, we are the first to propose a simple framework for training provably averaged (contractive) denoisers using unfolding networks.

We provide numerical results to validate our theoretical results and compare our algorithms with state-of-the-art regularization techniques. We also point out some future research directions stemming from the thesis.

Acknowledgements

I would have to express my sincere gratitude to a lot of people for helping me to perform the research which has gone into this dissertation.

Firstly, I would like to thank my advisor, Prof. Kunal Narayan Chaudhury for the continuous support of my Ph.D. study and related research, for his patience, motivation, and immense knowledge. Numerous technical discussions allowed me to better my thinking process and understand how to overcome my technical deficiencies. More importantly, he taught me to find connections between different fields of study which further resulted in the systematic flow between different projects I have been part of.

During my Ph.D. curriculum, I had taken a few courses which strengthened my research capabilities, so I convey my regards to the course instructors Prof. Gautam Bharali, Prof. P.S. Sastry, Prof. Venu Madhav Govindu, and Prof. Aditya Gopalan. I also thank Prof. Rajbabu Velmurugan and Prof. M. K. Bhuyan for carefully examining my thesis and providing valuable comments, which I believe has increased the quality of this dissertation.

This work was made possible because of various discussions with my lab-mates aka like-minded friends, especially Ruturaj Gavaskar, Unni V.S., Niladri Ranjan Das and Sanjay Ghosh. It has been a pleasure to work with all of them. In particular, the discussions with Ruturaj Gavaskar and Unni V.S. helped me to gain different insights about topics of our interest. I will never forget the tea sessions I had with all my labmates, where technical and non-technical rants were enjoyed by each other.

This journey was an exciting one because of the support of all my friends, so I would like to specially acknowledge Sudarshan Nayak, Prijesh K John, Saravana Bharathi, Preetham Samuel, Sethu, Vignesh. I would also like to extend my thanks to all the staff of the Electrical Engineering department who were helpful with all non-academic procedures.

I wish to thank my parents and my wife, Uma. I am certain that this thesis would not have been possible without their love, support, patience, and sacrifice. I dedicate this thesis to all of them.

Finally, I acknowledge the whole community of the Indian Institute of Science for giving me the opportunity to become a part of this esteemed institute. I also thank everyone who were responsible to develop such a beautiful research environment inside a lush green campus.

Pravin Nair

Publications

This thesis is based on the content from the following journal papers.

1. **Pravin Nair**, Ruturaj Gavaskar, and Kunal N. Chaudhury, "Fixed-point and objective convergence of plug-and-play algorithms," *IEEE Transactions on Computational Imaging (TCI)* vol. *7*, pp. 337–348, 2021.

2. **Pravin Nair**, and Kunal N. Chaudhury, "Plug-and-play regularization using linear solvers," *IEEE Transactions on Image Processing (TIP)* vol. 31, pp. 6344-6355, 2022.

3. **Pravin Nair**, and Kunal N. Chaudhury, "On the Construction of Averaged Deep Denoisers for Image Regularization," *https://arxiv.org/abs/2207.07321*, arxiv, 2022.

Other published/submitted papers, whose content does not form part of this thesis:
Journal Papers

1. **Pravin Nair** and Kunal N. Chaudhury, "Fast high-dimensional kernel filtering." *IEEE Signal Processing Letters* vol. 26, no. 2. pp. 377-381, 2019

   In this paper, the Nyström method is applied to develop a fast approximation algorithm for kernel filters like bilateral and nonlocal means. We also provide theoretical guarantees for the difference between our approximation and the original filter output.

2. Unni V. S., **Pravin Nair**, and Kunal N. Chaudhury, "Guided Nonlocal Patch Regularization and Efficient Filtering based Inversion for Multiband Image Fusion," *IEEE Transactions on Computational Imaging*, 2022.

   In this paper, we propose a convergent ADMM-based algorithm for hyperspectral image fusion by designing a nonlocal patch-based regularizer. Our algorithm is shown to outperform state-of-the-art variational as well as deep learning techniques.

3. Unni V. S., **Pravin Nair**, and Kunal N. Chaudhury, "Hyperspectral Fusion using Weighted Nonlocal Total Variation," *IEEE Geoscience Remote Sensing Letters,* 2022.

   In this paper, we design a regularizer based on nonlocal total variation for HS-MS fusion, where a multispectral image is used to calculate similarity weights. The proposed sparsity-promoting regularizer is shown to outperform existing regularization techniques.

4. Unni V. S., **Pravin Nair**, and Kunal N. Chaudhury, "Patch-based Regularzation for Pansharpening," submitted to *Signal Processing: Image Communication*.

   In this paper, we propose a fast patch-based regularization technique for pansharpening. The proposed algorithm is shown to outperform state-of-the-art methods, both visually and in terms of spatial and spectral quality measures.

Conference Papers

1. **Pravin Nair**, Ruturaj Gavaskar, and Kunal N. Chaudhury, "Compressive adaptive bilateral filtering," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2078-2082, 2020.

   In this paper, we propose a fast and efficient approximation for an adaptive variant of the bilateral filter, where the range kernel is allowed to vary from pixel to pixel. The speed-up obtained by the algorithm is due to the approximation of SVD using the Nyström method.

2. **Pravin Nair**, Unni V. S., and Kunal N. Chaudhury, "Hyperspectral Image Fusion using Fast High-dimensional Denoising," *IEEE International Conference on Image Processing*, pp. 3123-3127, September 2019.

   In this work, we propose a "plug-and-play" framework for hyperspectral image fusion. The algorithm is provably convergent and demonstrates excellent reconstruction capabilities.

3. Unni V. S., **Pravin Nair**, and Kunal N. Chaudhury, "Plug-and-Play Registration and Fusion," *IEEE International Conference on Image Processing*, pp. 2546-2550, October 2020.

   In this paper, we design a high-performance algorithm to synthetically fuse a high-spatial, low-spectral resolution image with a low-spatial, high-spectral resolution image to achieve high spatial and spectral resolution, even when the input images are misaligned.

Contents

List of Figures

List of Tables

Introduction

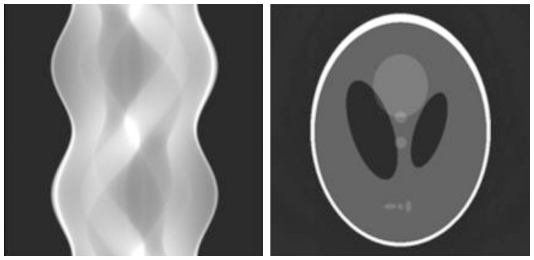
Many challenging image processing tasks can be posed as an inverse problem. Some well-known examples include deblurring [1], inpainting [2], compressed sensing [3], superresolution [4] and despeckling [5]. Such problems are typically ill-posed and hence the need for regularization. This thesis focuses on developing algorithms for these and other inverse problems, that leverage the power of modern denoisers for image regularization.

1.1 Imaging Inverse Problems

In several applications in science and technology, the recorded measurements are distorted or corrupted versions of a ground-truth image [6]. The observation can either be an image or some generic measurement. In general, the measurement process can often be captured by a mathematical model which describes the process by which the observations were obtained. In particular, if $\boldsymbol{x}_0 \in \mathbb{R}^n$ is the ground-truth image and the degradation process is modeled using an operator $\mathscr{H}$, then the observation $\boldsymbol{y} \in \mathbb{R}^m$ can be mathematically expressed as,

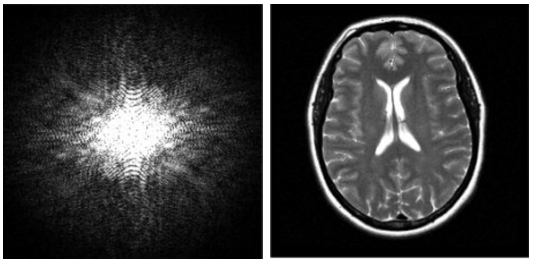
$$\boldsymbol{y} = \mathscr{H}(\boldsymbol{x}_0, \boldsymbol{\eta}), \tag{1.1}$$

where $\boldsymbol{\eta}$ is typically some random noise. In this thesis, $\mathscr{H}$ is assumed to be known or estimated. The objective is to reconstruct the underlying ground-truth image $\boldsymbol{x}_0$ from the observations $\boldsymbol{y}$ [6]. This abstract problem of inverting a given measurement model comes up in several computational imaging applications [7, 8].



(a) Sinogram. (b) X-ray CT image.

Figure 1.1: X-Ray Tomography: Sinogram consists of intensities measured along different incident angles.



(a) K-space. (b) MRI reconstruction.

Figure 1.2: MRI: Measurements are subsampled Fourier space measurements.

Many image processing tasks can be described by the model $y = \mathscr{H}(x_0, \eta) = \mathbf{A}x_0 + \eta$, where $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a linear operator and η is Gaussian noise. We call these as linear inverse problems: deblurring [1], inpainting [2], compressed sensing [3], and superresolution [4]. In tomography, compressed sensing and MRI, the incomplete noisy measurements $y \in \mathbb{R}^m$ are not images [6, 7] and $m \ll n$. On the other hand, in deblurring [1], inpainting [2] and superresolution [4], the observed measurements are indeed meaningful images. The forward model, for the above-mentioned inverse problems, are listed in Table 1.1.

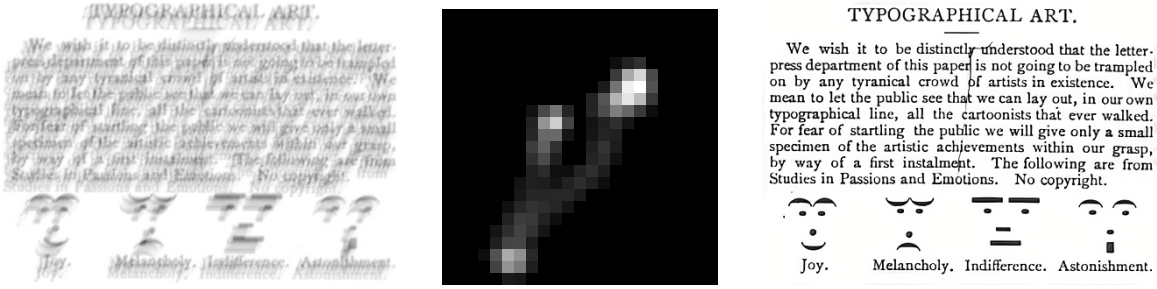
Table 1.1: Linear Inverse Problems

Application	Forward Model	Comments
Denoising [9]	$\mathbf{A} = \mathbf{I}$	$\mathbf{I}$ is the identity matrix
Deblurring [1]	$\mathbf{A} = \mathbf{B}$	$\mathbf{B}$ is a circulant matrix corresponding to a blur filter (for example Gaussian blur).
Inpainting [3]	$\mathbf{A} = \mathbf{S}$	$\mathbf{S}$ is a sensing matrix; an identity matrix with missing rows where every row corresponds to a single pixel.
Superresolution [4]	$\mathbf{A} = \mathbf{SB}$	$\mathbf{S}$ is decimation operator (identity matrix with missing rows) and $\mathbf{B}$ is circulant matrix corresponding to blur filter.
Compressive Sensing [3]	$\mathbf{A} = \mathbf{SF}$ or a random Gaussian or Bernouli matrix	$\mathbf{S}$ is a decimation matrix as in superresolution and $\mathbf{F}$ is Fourier transform matrix.
MRI [10]	$\mathbf{A} = \mathbf{SFD}$	$\mathbf{S}$ and $\mathbf{F}$ are as in Compressive Sensing. $\mathbf{D}$ is a diagonal matrix denoting spatial domain multiplication with coil sensitivity map.
CT [11]	$\mathbf{A} = \mathbf{R}$	$\mathbf{R}$ is discrete radon transform.

There exist inverse problems in imaging where $\mathscr{H}$ is nonlinear and the noise added is neither additive nor Gaussian. This includes applications such as single photon imaging [12], Poisson denoising [13], and despeckling [5]. For example, in synthetic aperture radar imaging [5], the measurements for a given pixel i is given by, $y(i) = x_0(i)\eta(i)$, where $x_0 \in \mathbb{R}^n$ is the unknown reflectance image and components of $\eta \in \mathbb{R}^n$ are iid Gamma random variables. On the other hand, in phase retrieval [14], $y = |\mathbf{A}x|^2$ where $\mathbf{A}$ is a complex measurement matrix and $|\cdot|^2$ is elementwise squaring. Two nonlinear inverse problems are shown in Figures 1.5 and 1.6.

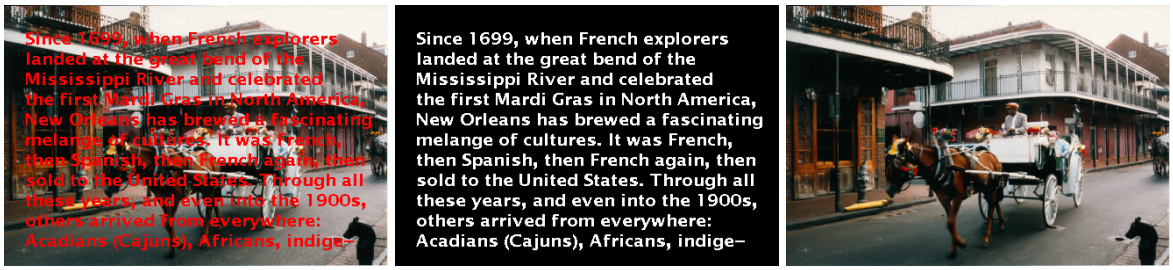
1.2 Image Regularization

Regularization is an important tool to develop algorithms for image reconstruction. Traditionally, image reconstruction problems have been solved using heuristic techniques. Some of these traditional methods include Gaussian filtering for image denoising, unsharp masking for image deblurring [15], interpolation techniques like bicubic downsampling for image superresolution [16], nonlinear partial differential equations for image inpainting [17]. However, these methods often do not produce aesthetic reconstructions due to the ill-posed nature of



(a) Blurred image. (b) Blur kernel. (c) Target.

Figure 1.3: Synthetic example for deblurring application where the ground-truth image is blurred using the filter in (b). It is difficult even for human beings to understand the lost information in (a).

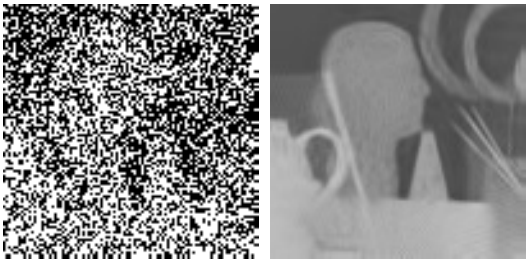


(a) Measurements. (b) Sensing mask. (c) Target.

Figure 1.4: Inpainting application where text in (b) is overlapped with the ground-truth image. This is a classic example of interpolating the missing pixel values.

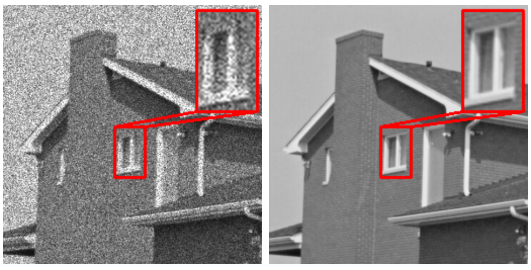
the problem. Ill-posed problems are essentially characterized by the existence of multiple solutions since the degradation model $\mathscr{H}$ is not invertible. Also, even when $\mathscr{H}$ is invertible, it can happen that a small change in measurements can lead to a large perturbation in the solution. To make this more precise, let us take the case of the linear inverse problem under no-noise conditions, where the forward model is $\boldsymbol{y} = \mathbf{A}\boldsymbol{x}_0$. We could have cases where $\mathbf{A}$ has a non-trivial nullspace and hence many solutions exist or even if $\mathbf{A}$ is invertible, the matrix has a high condition number making the solution sensitive to even small changes in measurement. Hence, we need to constrain the solution (search) space for reconstructions. In particular, force the reconstruction to resemble the ground-truth image (class of natural images), which is essentially known as regularization.

Recall the generic forward model $\boldsymbol{y} = \mathscr{H}(\boldsymbol{x}_0, \boldsymbol{\eta})$ for the measurements in the inverse problems. Note that in all the inverse problems we discussed, the assumption that the measurements are random is valid due to the noise component. In particular, we can model measurements using the likelihood function $p(\boldsymbol{y}|\boldsymbol{x})$. For example, take the case of linear inverse problems, where noise $\boldsymbol{\eta}$ is additive Gaussian i.e. $\boldsymbol{\eta} \sim \mathcal{N}(0, \mathbf{I})$. Then, given the ground-truth image $\boldsymbol{x}_0$ the measurement $\boldsymbol{y}$ is also Gaussian; $\boldsymbol{y} \sim \mathcal{N}(\mathbf{A}\boldsymbol{x}_0, \mathbf{I})$. The Maximum Likelihood Estimate



(a) Measurements. (b) Target.

Figure 1.5: Single Photon Imaging where measurements are captured using uniformly distributed multiple photon detectors.



(a) Measurements. (b) Target.

Figure 1.6: Removing multiplicative noise from the measurements for despeckling; commonly occur in Synthetic Apperture Radar (SAR) imaging.

(MLE) is the solution of the following problem:

$$\max_{x\in\mathbb{R}^n} \quad p_{y|x}(y|x), \tag{1.2}$$

which finds the reconstruction estimate that maximizes the likelihood of observed measurements. For the special case of linear inverse problems, taking negative log-likelihood, we can reduce (1.2) to the classical least square problem $\min_{x\in\mathbb{R}^n} ||y-\mathbf{A}x||^2$. One of the very early techniques, Landweber method [18], solves this using gradient descent. Moreover, running the gradient descent algorithm for a few iterations is shown to regularize the reconstruction output. Note that MLE estimates do not incorporate any prior information about the image to be reconstructed and hence the reconstruction need not be close to the ground-truth image (eg. see Fig. 1.7).

1.2.1 Bayesian Prior

In the Bayesian setting, our belief about the reconstruction image can be modeled using some prior distribution $p_x(x)$ on the ground truth. Further, the likelihood term can be modeled using the conditional distribution $p_{y|x}(y|x)$ derived from the forward model. The posterior distribution $p_{x|y}(x|y)$ of the reconstruction x models the conditional distribution of the estimate x given the measurements y. Bayes' theorem provides a convenient way of computing this from the likelihood and prior.

$$p_{x|y}(x|y) = \frac{p_{y|x}(y|x)p_x(x)}{\int\ p_{y|x}(y|x)p_x(x)\,\mathrm{d}x}.$$

The Maximum A Posteriori (MAP) estimate of the ground truth is given by the solution of $\max_x p_{x|y}(x|y)$, or equivalently by solving

$$\max_{x\in\mathbb{R}^n} \quad p_{y|x}(y|x)p_x(x). \tag{1.3}$$

Taking negative log likelihood and by changing maximization to minimization, we can express (1.3) as,

$$\min_{x\in\mathbb{R}^n} \quad f(x)+g(x), \tag{1.4}$$

where $f(x) \propto -\log(p_{y|x}(y|x))$ and $g(x) \propto -\log(p_x(x))$. Intuitively, the better the prior about the ground truth, the closer we expect the reconstruction to be to the ground truth. So how do we model the prior $p_x(x)$? To understand this, let us take the case of linear inverse problems, where noise $\eta \sim \mathcal{N}(0, \sigma_n^2\mathbf{I})$ and hence the measurement density function is $y|x \sim \mathcal{N}(\mathbf{A}x, \mathbf{I})$. We can assume the prior also to have Gaussian density $\sim \mathcal{N}(0, \sigma_0^2\mathbf{I})$. Then, the optimization problem in (1.4) reduces to solving,

$$\min_{x\in\mathbb{R}^n} \quad \frac{1}{2}||y-\mathbf{A}x||^2 + \frac{1}{2\sigma_0^2}||x||^2. \tag{1.5}$$

1.2.2 Energy Minimization

Note that finding MAP estimate in (1.4) is very restrictive as the term $g(x)$ is the log of prior density, where prior information needs to be modeled. Instead of modeling the prior density, we can directly consider minimizing the energy functional, in effect solving the following optimization problem.

$$\min_{x\in\mathbb{R}^n} \quad f(x)+\lambda g(x), \tag{1.6}$$

where λ is the regularization parameter, used to balance the data-fidelity term f and regularizer g in (1.6) (we might sometimes absorb λ in the data-fidelity term for convenience). In particular, the energy functional $f+\lambda g$ is composed of the data fidelity term f and a cleverly chosen regularizer $g : \mathbb{R}^n \to \mathbb{R} \cup \infty$. In linear inverse problems [19, 20], since the conditional density function for the measurements y is Gaussian ($y|x \sim \mathcal{N}(\mathbf{A}x, \mathbf{I})$), the log- likelihood or data-fidelity term f is given by $f(x) \propto -\log(\exp(||y-\mathbf{A}x||^2)) = ||\mathbf{A}x-y||^2$ i.e. f is quadratic and convex. On the other hand, the data-fidelity term is non-quadratic but convex in problems such as single photon imaging [12], Poisson denoising [13], and despeckling [5].

The regularizer g should penalize the reconstruction estimate x that looks significantly different from natural images; this in effect forces the reconstruction to resemble the ground truth. Due to computational considerations and for guaranteeing the existence and uniqueness of solutions to (1.6), g is often taken to be convex [8]. Two of the commonly used regularizers are total variation (TV) and wavelet-based regularizers. TV regularizer [21], is given by:

$$\sum_{i=1}^{n} |\partial_h x_i| + |\partial_v x_i|, \tag{1.7}$$

where $\partial_h x_i$ and $\partial_v x_i$ give horizontal and vertical components of discrete gradient at a pixel x_i. The total-variation regularizer penalizes the images which have more edges, unlike natural

images. Another commonly used energy-based model includes wavelet-based regularizer which is of the form $||\mathbf{W}\boldsymbol{x}||_1$, where $\mathbf{W}$ is wavelet matrix and $||\cdot||_1$ denotes ℓ_1 norm. The wavelet-based regularizer promotes images which are sparse in the wavelet transformed domain [22]. Note that we will be using wavelet-based regularizer in Chapter 4.

In Fig. 1.7, we show that solving the problem in (1.6) with TV regularizer in (1.7), instead of MLE in (1.2) indeed provides an improved estimate for the superresolution application. Notice that even after using TV regularization, the image reconstructed is nowhere close to the ground-truth image. So, understanding the distribution of natural images and coming up with handcrafted regularizers promoting certain properties for reconstruction estimates is an ongoing research problem.

(a) Ground-Truth $\boldsymbol{x}_0$. (b) $\boldsymbol{x}_{\text{MLE}}$. (c) TV regularization.

Figure 1.7: Energy function minimization using TV regularizer shows an improved performance (PSNR of 22dB) as compared to MLE (PSNR of 19dB) for superresolution application where $\boldsymbol{x}_0$ is downsampled by a factor 3 to generate measurement $\boldsymbol{y}$.

The construction of a handcrafted regularizer g for imaging inverse problems has been the topic of active research in the last few decades. There have been several regularizers proposed over the years. This includes Tikhonov regularizer (the regularizer term in (1.5)) [23], total-variation regularizer [21] and its variants [24, 25], entropy maximization [26], sparsity-based regularizers [27, 28], Markov random fields [29], Hessian regularizer [30], GMM regularizer [31], graph Laplacian [32], patch-based regularizers like K-SVD [33], weighted nuclear norm minimization [34] etc. We refer [6, 7, 8, 35] for more details about regularizers for image reconstruction.

1.3 Iterative Minimization

We first need to check for the existence (and uniqueness) of the solutions for the optimization problem in (1.6), which further depends on f and g. For example, if f and g are convex, the conditions for existence of solutions are widely known [36]. We refer [37] for extensive results on existence and uniqueness of solutions for (1.6) for various assumptions on f and g. Further, if the solution exists, how do we compute it? The first preferred algorithm is

gradient descent, where, starting with $x_0 \in \mathbb{R}^n$, the updates are given by

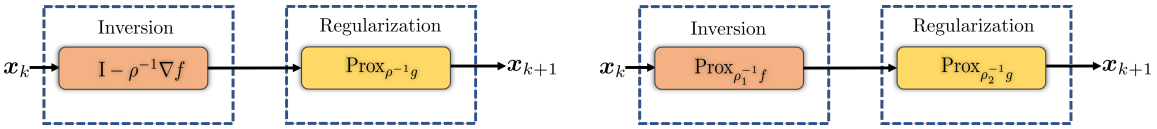
$$x_{k+1} = x_k - \rho\big(\nabla f(x_k) + \nabla g(x_k)\big), \tag{1.8}$$

where $\rho > 0$ is the step-size. However, we require both the data-fidelity term $f(x)$ and the regularizer $g(x)$ to be differentiable in this case. Hence, we cannot use gradient descent to solve (1.6) if g is total variation regularizer in (1.7) or wavelet-based regularizer, since these regularizers are non-differentiable. In fact, we need to move away from gradient-based algorithms since most regularizers are non-differentiable.

The widely used algorithms to solve (1.6) for general choices of regularizer g are termed as proximal algorithms [38], in which the regularization is performed using the proximal operator of g:

$$\text{Prox}_g(x) = \underset{z\in\mathbb{R}^n}{\text{argmin}} \quad \frac{1}{2}||z-x||^2 + g(z), \tag{1.9}$$

where $||\cdot||$ is the Euclidean norm. Note that to calculate $\text{Prox}_g(x)$, g need not be differentiable. In fact, this is well-defined for any closed, proper and convex g [38]. For example, let $g(x) = ||Wx||_1$ where W is an orthogonal wavelet transform. The regularizer g is non-differentiable but $\text{Prox}_g(x)$ exists for every x and has closed form solution [39].



(a) ISTA. (b) HQS.

Figure 1.8: $x_k \to x_{k+1}$ update in ISTA and HQS.

The class of proximal algorithms include Iterative Shrinkage Thresholding Algorithm (ISTA) [40], Alternating Direction on Method of Multipliers (ADMM) [41], Douglas Rach-Ford Splitting (DRS) [42], Half Quadratic Splitting (HQS) [43] etc. These iteration schemes in effect "split" the data-fidelity term and regularization term in every iteration, and are hence referred to as splitting algorithms as well. Every proximal algorithm has two main steps: an inversion step that involves the data-fidelity term f, and a regularization step in the form of proximal operation. In this thesis, we will focus only on some commonly used proximal algorithms in imaging — ISTA, ADMM, and HQS. For example, the ISTA update $x_k \to x_{k+1}$ is given by,

$$\begin{aligned} \text{(inversion)}: &\quad z_{k+1} = x_k - \rho^{-1}\nabla f(x_k), \\ \text{(regularization)}: &\quad x_{k+1} = \text{Prox}_{\rho^{-1}g}(z_{k+1}). \end{aligned} \tag{1.10}$$

where $\rho > 0$ is the step-size and $x_0 \in \mathbb{R}^n$ is an arbitrary initialization. Note that ISTA requires

$f(\boldsymbol{x})$ to be differentiable. If $f(\boldsymbol{x})$ is non-differentiable, we can use HQS, where starting with $\boldsymbol{x}_0 \in \mathbb{R}^n$, the update $\boldsymbol{x}_k \to \boldsymbol{x}_{k+1}$ is performed as,

$$\begin{aligned} \text{(inversion)}: \quad & \boldsymbol{z}_{k+1} = \text{Prox}_{\rho_1^{-1} f}(\boldsymbol{x}_k), \\ \text{(regularization)}: \quad & \boldsymbol{x}_{k+1} = \text{Prox}_{\rho_2^{-1} g}(\boldsymbol{z}_{k+1}), \end{aligned} \tag{1.11}$$

where $\rho_1, \rho_2 > 0$ are intrinsic parameters. Similarly, we can also use ADMM, where starting with $\boldsymbol{x}_0, \boldsymbol{z}_0 \in \mathbb{R}^n$, the updates are, given by:

$$\begin{aligned} \text{(inversion)}: \quad & \boldsymbol{v}_{k+1} = \text{Prox}_{\rho^{-1} f}(\boldsymbol{x}_k - \boldsymbol{z}_k), \\ \text{(regularization)}: \quad & \boldsymbol{x}_{k+1} = \text{Prox}_{\rho^{-1} g}(\boldsymbol{v}_{k+1} + \boldsymbol{z}_k), \\ \text{(dual update)}: \quad & \boldsymbol{z}_{k+1} = \boldsymbol{z}_k + (\boldsymbol{v}_{k+1} - \boldsymbol{x}_{k+1}), \end{aligned} \tag{1.12}$$

The updates for ISTA are typically cheaper than ADMM updates in (1.12) or HQS updates in (1.11), since, in general, the gradient update has lesser computational complexity than proximal operation. In ADMM and HQS, the data-fidelity term need not be differentiable unlike ISTA, but $\text{Prox}_{\rho^{-1} f}$ should be easily computable in closed-form or iteratively. All the three proximal algorithms ISTA, HQS, and ADMM can be expressed as dynamical system of the form $\boldsymbol{x}_{k+1} = \text{T}(\boldsymbol{x}_k)$ for some operator T. For example, the dynamical system corresponding to ISTA and HQS are shown in Figures 1.8. We will use this property of proximal algorithms to great effect in the upcoming chapters.

Having sketched out the flow of developing regularizer-based optimization schemes for solving inverse problems, we now discuss key points regarding such optimization-based solutions. The advantages of regularization-based optimization are:

- Clear interpretation as an inversion step followed by regularization is iterated in a feedback loop. It can be proven that the reconstruction gets closer to the ground-truth image as iterations increases, provided we choose the right parameters.

- Convergence of ISTA, ADMM, and HQS are well established when f and g are convex which is the case for many imaging applications. The convergence guarantees for ISTA and ADMM can be proved using multiple mathematical techniques. In Sections 4.2 and 4.3, we resort to the method of realizing ISTA and ADMM iterations as averaged operators to provide convergence guarantees.

- Flexibility in handling various image restoration tasks by simply specifying $\mathscr{H}$, noise model $\boldsymbol{\eta}$ and degraded measurements $\boldsymbol{y}$.

On the other hand, the drawbacks are:

- Designing hand-crafted regularizers g is a difficult task. If the regularizers are sophisticated, the proximal operation may not have a closed-form. The need for sophisticated regularizers is clear from the example in Fig. 1.7.

- The parameters for the algorithms are user-dependent and choosing the parameters is a tedious task, mostly done by trial and error.

- Model-based optimization using hand-crafted regularizers is either weaker than deep learning methods (which is discussed next) or the optimization process is time-consuming [35].

1.4 Learning Based Models

Currently, the most popular method to solve the inverse problems is using deep neural networks [44]. There are many different methods where deep neural networks are used for image reconstruction—denoising autoencoders [45], U-Net [46], deep convolutional networks [47], deep unfolding networks [48], Neumann networks [49] and deep image prior [50]. We will focus only on two supervised learning models in this thesis: convolutional neural networks (CNN) and deep unfolding networks (DUN). The key idea here is to regress a parametric function class over the data samples generated by the forward model $\boldsymbol{y} = \mathscr{H}(\boldsymbol{x}_0, \boldsymbol{\eta})$. In effect, the inverse of operator $\mathscr{H}$ is approximated by a parametrized function, where the parameters are learnt.

The learning problem is posed as follows. Let the clean images (or patches extracted from the ground truth) be $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_M \in \mathbb{R}^p$, where p is the number of pixels. The degraded images $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M \in \mathbb{R}^p$ are generated by corrupting $\boldsymbol{z}_i$ using the measurement model, i.e., $\boldsymbol{y}_i = \mathscr{H}(\boldsymbol{z}_i, \boldsymbol{\eta}_i)$, where $\boldsymbol{\eta}_i$ is randomly sampled from the noise distribution. The goal is to train a network $\mathrm{N} : \mathbb{R}^p \to \mathbb{R}^p$ such that $\boldsymbol{z}_i \approx \mathrm{N}(\boldsymbol{y}_i; \Theta)$, where, Θ is the set of parameters. This is done by solving the optimization problem.

$$\min_{\Theta} \quad \frac{1}{M} \sum_{i=1}^{M} \ell(\mathrm{N}(\boldsymbol{y}_i; \Theta), \boldsymbol{z}_i), \tag{1.13}$$

where $\ell(\cdot, \cdot)$ is some loss function. We will next discuss the following models for inverse problems are a) Convolutional Neural Networks and b) Deep Unfolding Networks.

1.4.1 Convolutional Neural Networks

We will start with an example to understand how N can be parametrized as CNN-based network for the application of superresolution from [9] (referred to as DnCNN). Recall that the forward model for the problem is given as $\boldsymbol{y} = \mathrm{S}\boldsymbol{x}$, where S is a bicubic downsampling operation. Given a set of patches $\{\boldsymbol{z}_i\}$ and corresponding upsampled patches $\{\boldsymbol{y}_i\}$, where

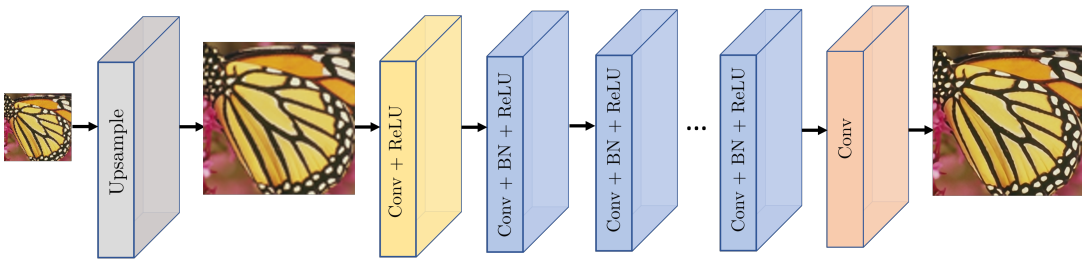


Figure 1.9: End-to-end CNN-based network for superresolution. Conv stands for convolutional layer, BN for batch normalization layer, and ReLU for the nonlinear activation function - rectified linear units. A + B implies B ∘ A. For example, Conv + ReLU implies convolution operation followed by ReLU activation function.

$\boldsymbol{y}_i = S\boldsymbol{z}_i$, the network is trained to satisfy $\boldsymbol{y}_i \approx N(\hat{\boldsymbol{z}}_i)$, where $\hat{\boldsymbol{z}}_i$ is bicubic interpolation of $\boldsymbol{z}_i$. The parametric form of N, used in [9], consists of a composition of transforms, as in,

$$N(\boldsymbol{y};\Theta) = K_L(R(B_{L-2}(K_{L-1}(\cdots(R(B_1(K_2(R(K_1(\boldsymbol{y})))))\cdots), \tag{1.14}$$

where

- $\boldsymbol{y} \in \mathbb{R}^{h\times w\times C_0}$ is the input image with C_0 channels.
- L is the number of convolutional layers.
- C_ℓ is the number of filters in convolutional layer $\ell \in \{1,\dots L\}$, which is also the number of channels in the output of layer ℓ; $C_\ell = 64$ for $\ell \in \{1,\dots,L-1\}$ and $C_L = C_0$.
- For $\ell \in \{1,2,\dots,L\}$, the filters in convolutional layer ℓ are of size $3\times 3\times C_{\ell-1}$.
- Convolutional layer ℓ is represented as a transform $K_\ell : \mathbb{R}^{h\times w\times C_{\ell-1}} \to \mathbb{R}^{h\times w\times C_\ell}$.
- BN layer $\ell \in \{1,\dots,L-2\}$ is represented as a transform $B_\ell : \mathbb{R}^{h\times w\times C_{\ell+1}} \to \mathbb{R}^{h\times w\times C_{\ell+1}}$.
- $R(t) = \max(t,0)$ is the output of the ReLU function which is applied component-wise.
- The parameter set Θ is the weights of the convolutional layers and the weights and biases of the BN layers.
- $N(\boldsymbol{y};\Theta) \in \mathbb{R}^{h\times w\times C_0}$ is the output from N with input $\boldsymbol{y}$ and hyperparameter setting Θ.

Note that in [9], separate networks are trained for downsampling factors such as 2, 3, and 4. Similarly CNNs are trained for various applications such as superresolution [51, 52], deblurring [53], compressed sensing [54], computed tomography [55], despeckling [56], poisson denoising [57] etc. We refer [47] for an extensive review of the recent uses of convolutional neural networks (CNNs) to solve inverse problems in imaging.

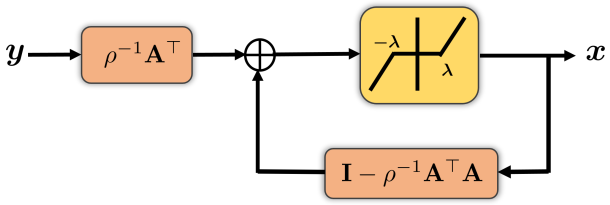


Figure 1.10: ISTA iteration for sparse coding.

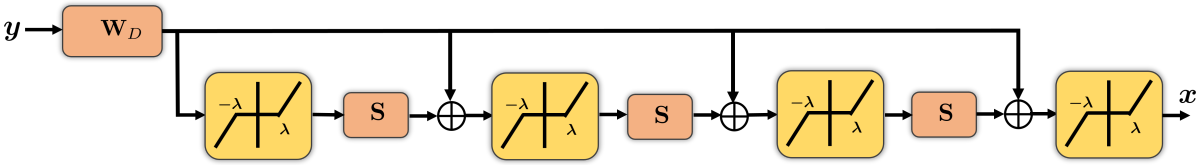


Figure 1.11: Unfolded ISTA network for sparse coding. Instead of learning $\mathbf{A}$ in Fig. 1.10, the idea is to learn the matrices $\mathbf{W}_D$ and $\mathbf{S}$ instead of $\rho^{-1}\mathbf{A}^\top$ and $\mathrm{I}-\rho^{-1}\mathbf{A}^\top\mathbf{A}$ respectively. These matrices are shared across iterations and these iterations are unfolded.

1.4.2 Deep Unfolding Networks

End-to-end CNN-based methods lack interpretation as we do not understand how exactly the regularization takes place. Inspired by model-based optimization, where the regularization is via a proximal operation, another common approach to learn a deep network is as follows [58]. 1) Choose any proximal algorithm. 2) Unfold the iterations and turn it into a network, where every iteration is treated as a layer. 3) Learn the algorithm-specific parameters.

The main idea in constructing deep unfolding networks is to interpret a truncated optimization algorithm like ISTA or ADMM (with a fixed number of iterations) as an end-to-end trainable deep network, where learnable parameters are intrinsic parameters of the algorithm, and every iteration is interpreted as a layer. We will give an example from [48] for the application of sparse coding. This is perhaps the first algorithm to train a network by unfolding ISTA iterations. The forward model for sparse coding is given by $\boldsymbol{y}=\mathbf{A}\boldsymbol{x}+\boldsymbol{\eta}$, where $\mathbf{A}$ is a dictionary matrix and $\boldsymbol{\eta}$ is Gaussian noise. Thus, application of sparse coding requires us to provide an estimate of $\boldsymbol{x}$ which is sparse (a smaller number of non-zero elements). Hence the chosen regularizer should promote sparse vectors; one such regularizer is given by $g(\boldsymbol{x})=||\boldsymbol{x}||_1$, where $||\cdot||_1$ is ℓ_1 norm. Then the optimization problem to be solved is of the form,

$$\underset{\boldsymbol{x}}{\operatorname{argmin}} \quad \frac{1}{2}||\boldsymbol{y}-\mathbf{A}\boldsymbol{x}||^2+\lambda||\boldsymbol{x}||_1, \tag{1.15}$$

where λ is the regularization parameter. Applying ISTA algorithm to solve (1.15) (shown as a feedback loop in Fig. 1.10), the update $\boldsymbol{x}_k \rightarrow \boldsymbol{x}_{k+1}$ is given as,

$$\begin{aligned}\boldsymbol{z}_{k+1} &= \boldsymbol{x}_k-\rho^{-1}(\mathbf{A}^\top\mathbf{A}\boldsymbol{x}_k-\mathbf{A}^\top\boldsymbol{y}),\\ \boldsymbol{x}_{k+1} &= S_\lambda(\boldsymbol{z}_{k+1}),\end{aligned} \tag{1.16}$$

where $S_\lambda(\cdot)$ is the proximal map of regularizer g, given by component-wise shrinkage operation ($= \boldsymbol{x} - \text{sign}(x)\lambda$ if $|x| > \lambda$ else zeros).

The authors in [48] fixed the number of iterations as $M = 3$ and the shrinkage parameter for soft thresholding. Further, the authors showed that ISTA for 3 iterations can be unfolded and the required matrices (known as dictionaries) can be learnt. Note that the shrinkage parameter for soft-thresholding can also be learnt if required. In particular, let $\mathbf{S}$ and $\mathbf{W}_d$ be the matrices to be estimated, which is indeed the approximation of $\mathrm{I} - \rho^{-1}\mathbf{A}^\top\mathbf{A}$ and $\rho^{-1}\mathbf{A}^\top$. Then N can be parametrized as shown in Fig. 1.11 with parameters $\Theta = \{\mathbf{S}, \mathbf{W}_d, \lambda\}$. The parameters are estimated as in the case of CNN-based methods using pairs of $(\boldsymbol{z}_i, \boldsymbol{y}_i)$, where $\boldsymbol{y}_i$ is some sparse representation of image $\boldsymbol{z}_i$.

We refer to such kinds of networks as deep unfolding networks. Note that for every unfolding network, we need to specify the proximal algorithm. In recent years, there is an increased interest in training deep unfolding networks for various applications, eg. ISTA has been unfolded for compressed sensing [58], MRI reconstruction [59]. HQS has been unfolded for superresolution [60]. ADMM was unfolded for CS-MRI [61], deblurring [62]. In all these works, the deep unfolding networks consistently outperform the corresponding proximal algorithms.

The advantages of using deep networks for solving inverse problems are as follows:

- Reconstruction quality is better than that of model-based optimization algorithms.
- Inference time is less since just a forward pass is required.
- During inference, no user input parameters are required to be tuned, unlike model-based optimization.

One of the drawbacks of using CNNs for solving inverse problems is the lack of interpretability. For example, it is not clear how the regularization is performed. In the case of deep unfolding networks, this is not the case, since the layers of the networks are indeed obtained by unfolding proximal algorithms. On the other hand, there are two clear drawbacks to training end-to-end-networks for imaging inverse problems. One drawback is that training end-to-end networks require sufficient corpus of clean data for the problem under consideration. This is not the case for problems in medical image reconstruction [63], though this issue can partly be solved using transfer learning [64].

The main drawback of end-to-end solutions is that they are trained with a fixed forward model. Thus, for different forward models and different noise distributions, we need to retrain the network from scratch. We illustrate this point with an example. Let us consider VDSR [52] for superresolution, where $\mathbf{S}$ is bicubic upsampling as in the experiment in Fig. 1.7. In this case, the output from VDSR in Fig. 1.12(c) is close to ground truth in Fig. 1.12(a) and shows massive improvement in reconstruction performance over bicubic interpolation in

(a) Ground-Truth. (b) S-bicubic. (c) VDSR output. (d) S-decimation. (e) VDSR output.

Figure 1.12: A minor change in the sampling operator in the forward model leads to minor perturbation in the input image to the VDSR network. Though VDSR is trained for superresolution, the network is not robust to the changes in the forward model for superresolution. VDSR is trained, by fixing the forward model $\boldsymbol{y} = \mathbf{S}\boldsymbol{x}$ with $\mathbf{S}$ as the bicubic upsampling operator. Thus if change the forward model where $\mathbf{S}$ is a decimation operation, VDSR network outputs a worse estimate than even the traditional bicubic upsampling method.

Fig. 1.12(b). Now we change the forward model by choosing $\mathbf{S}$ as the decimation operation. Then the accuracy (measured using PSNR) of VDSR output in Fig. 1.12(e) is worse than bicubic interpolation in Fig. 1.12(d) by 3 dB. Thus, VDSR is not robust to changes in the forward model.

1.5 Regularization using Denoising

Recently, Plug-and-Play (PnP) and Regularization by Denoising (RED) have garnered wide interest in image reconstruction applications. These algorithms are motivated by model-based optimization and the existence of high-performance Gaussian denoisers. We first give a brief survey on Gaussian denoisers and then explain PnP and RED.

1.5.1 Gaussian Denoisers

Image denoising problem, and in particular, Gaussian denoising, has widely been studied in the past several decades. The forward model for denoising is given by $\boldsymbol{y} = \boldsymbol{z} + \boldsymbol{\eta}$, where $\boldsymbol{y} \in \mathbb{R}^p$ is the noisy measurements, $\boldsymbol{z} \in \mathbb{R}^p$ is the ground-truth patch, $\boldsymbol{\eta}$ is Gaussian noise of standard deviation σ and p is number of pixels in the image. Similar to restoration, we could come up with optimization models for denoising (referred to as variational denoisers) by choosing a regularizer g and solving

$$\hat{\boldsymbol{x}} = \underset{x}{\operatorname{argmin}} \quad \frac{1}{2}||\boldsymbol{y} - \boldsymbol{x}||^2 + \sigma^2 g(\boldsymbol{x}), \tag{1.17}$$

Note that, by the definition of proximal operator in (1.9), $\hat{\boldsymbol{x}}$ is simply $\text{Prox}_{\sigma^2 g}(\boldsymbol{y})$. Recall that, in model-based optimization, the update $\boldsymbol{x}_k$ in ISTA (1.10), ADMM (1.12), HQS (1.11), etc. is indeed variational denoising for some regularizer g. Most widely used variational regularizers include Total Variation (TV) [21] and its extensions [24, 25], Hessian based [30], sparsity promoting [27, 28], Gaussian Mixture Model based [31], graph Laplacian based [32], patch-based regularizers like K-SVD [33], weighted nuclear norm minimization (WNNM)

[34] etc.

On the other hand, there are also high-quality image denoisers (both linear and nonlinear) that are not based on the variational minimization. For example, kernel denoisers such as [65], Lee filter [66], bilateral filter [67], nonlocal means (NLM) [68], LARK [69] provide impressive denoising performance but are computational expensive. These can generally be expressed as

$$\mathrm{D}(\boldsymbol{x})_i = \frac{\sum_{j=1}^{n} \omega_{ij} \boldsymbol{x}_j}{\sum_{j=1}^{n} \omega_{ij}}, \tag{1.18}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the noisy input image, $\mathrm{D}(\boldsymbol{x})$ is the denoised output, $\boldsymbol{x}_i$ denotes the intensity value of $\boldsymbol{x}$ at the location $\boldsymbol{i} \in [1, n]$, and ω_{ij} measures the similarity between pixels corresponding to indices i and j in the image $\boldsymbol{x}$. If ω_{ij} does not depend on the input image $\boldsymbol{x}$, for example, if the similarity measure ω_{ij} is computed using a guide image, then the denoiser in (1.18) is indeed linear. In particular, the non-local means filter, when weights are computed from a fixed surrogate image, is a linear denoiser and provide impressive denoising performance.

State-of-the-art denoisers are however nonlinear and mostly learning-based. The best non-learning-based methods are a collaborative filtering approach BM3D [70] and a variational based approach WNNM [34]. Most widely used learning-based denoisers include TNRD [71], DnCNN [9], IRCNN [72], FFD-NET [73], MWCNN [74], NLRN [75], DRU-NET [76].

1.5.2 Plug-and-Play Algorithms

In Plug-and-Play regularization (or simply PnP), instead of having to handcraft the regularizer g and apply it via its proximal operator, the authors in [77] proposed to directly "plug" a powerful denoiser in proximal algorithms. To make the idea of PnP algorithms precise, we will consider ISTA. Recall that to solve the(1.6), the ISTA update $\boldsymbol{x}_k \to \boldsymbol{x}_{k+1}$ is given by,

$$\boldsymbol{z}_{k+1} = \boldsymbol{x}_k - \rho^{-1} \nabla f(\boldsymbol{x}_k), \quad \boldsymbol{x}_{k+1} = \mathrm{Prox}_{\rho^{-1} g}(\boldsymbol{z}_{k+1}). \tag{1.19}$$

In every classical proximal algorithm like ISTA, we noted that the regularization is performed in the form of the proximal operator of regularizer g ($\boldsymbol{x}$ update in (1.19)). Suppose we choose prior density $p(\boldsymbol{x}) = \exp(-\rho^{-1} g(\boldsymbol{x}))$, assume the measurements are $\boldsymbol{z}_{k+1}$, and set the likelihood function to be Gaussian with variance ρ^{-1}. Then, $\mathrm{Prox}_{\rho^{-1} g}(\boldsymbol{z}_{k+1})$ is indeed the MAP estimation problem for denoising as in (1.3). This observation is the exact motivation to replace $\mathrm{Prox}_{\rho^{-1} g}$ in (1.10) with D, where D can be chosen as any arbitrary Gaussian denoiser [77]. Note that the denoiser D need not be variational based such as total variational or wavelet denoising. In particular, starting with $\boldsymbol{x}_0 \in \mathbb{R}^n$, the updates for the PnP variant of the

parent algorithm ISTA, referred to as PnP-ISTA are:

$$z_{k+1} = x_k - \rho^{-1}\nabla f(x_k), \quad x_{k+1} = \mathrm{D}(z_{k+1}). \tag{1.20}$$

Similarly, in PnP-ADMM, starting with $x_0, z_0 \in \mathbb{R}^n$, the updates are:

$$\begin{aligned} v_{k+1} &= \mathrm{Prox}_{\rho^{-1}f}(x_k - z_k), \\ x_{k+1} &= \mathrm{D}(v_{k+1} + z_k), \\ z_{k+1} &= z_k + (v_{k+1} - x_{k+1}), \end{aligned} \tag{1.21}$$

We refer to [78, 72, 79, 80, 81, 82] for PnP variants of HQS, DRS, and AMP, and online variants of PnP-ISTA and PnP-ADMM.

(a) Ground-Truth. (b) TV (PSNR: 24dB). (c) CNN (PSNR: 27dB).

Figure 1.13: Using DnCNN instead of Total Variational Denoising in PnP-ISTA shows improved performance of around 3dB. Edges are clearer and textures are not smoothed out in (c) as compared to (b).

Remarkably, PnP has been shown to produce promising results across a wide range of applications including superresolution, MRI, fusion, and tomography [78, 72, 45, 79, 81, 83, 84, 85, 86, 87, 88, 89, 90]. Most of the best performing PnP algorithms like [72, 76, 87] use trained deep networks like DnCNN [9], IRCNN [72] etc, thereby incorporating the advantages of model-based optimization methods and learning-based denoisers. A sample result is shown in Fig. 1.13, where choosing D as a high-performance CNN-based denoiser instead of classical total variational denoising in (1.20) is shown to obtain better regularization. Note that applying total variational denosing in (1.20) is same as selecting total variation regularizer in (1.10).

1.5.3 Regularization by Denoising

In Regularization by Denoising (or simply RED), explicit regularizers based on the denoisers are constructed. For example, in the original paper [91], the authors proposed to work with

the regularizer

$$\frac{1}{2}\boldsymbol{x}^\top\big(\boldsymbol{x}-\mathrm{D}(\boldsymbol{x})\big), \tag{1.22}$$

and the corresponding reconstruction problem

$$\min_{x} \quad f(\boldsymbol{x})+\frac{\lambda}{2}\boldsymbol{x}^\top\big(\boldsymbol{x}-\mathrm{D}(\boldsymbol{x})\big), \tag{1.23}$$

where D is any arbitrary denoiser and λ is the regularization parameter. This is solved iteratively using ADMM or gradient-based methods, where the gradient of regularizer in (1.22) is approximated by $\boldsymbol{x}-\mathrm{D}(\boldsymbol{x})$ [91]. In particular, for gradient descent RED, the update $\boldsymbol{x}_k \to \boldsymbol{x}_{k+1}$ is given by:

$$\boldsymbol{z}_{k+1}=\mathrm{D}(\boldsymbol{x}_k), \quad \boldsymbol{x}_{k+1}=\boldsymbol{x}_k-\mu\big(\nabla f(\boldsymbol{x}_k)+\lambda(\boldsymbol{x}_k-\boldsymbol{z}_{k+1})\big),$$

where $\mu,\lambda>0$. It was later shown that this gradient of $\boldsymbol{x}-\mathrm{D}(\boldsymbol{x})$ is in fact exact for the regularizer when the denoiser D has a symmetric Jacobian and satisfies local homogeneity [92]. RED has shown state-of-the-art performance for phase retrieval [93, 94]. Variants of RED have been applied for tomography [95], superresolution and deblurring [96], compressed sensing MRI [97], deblurring and compressed sensing [98] etc.

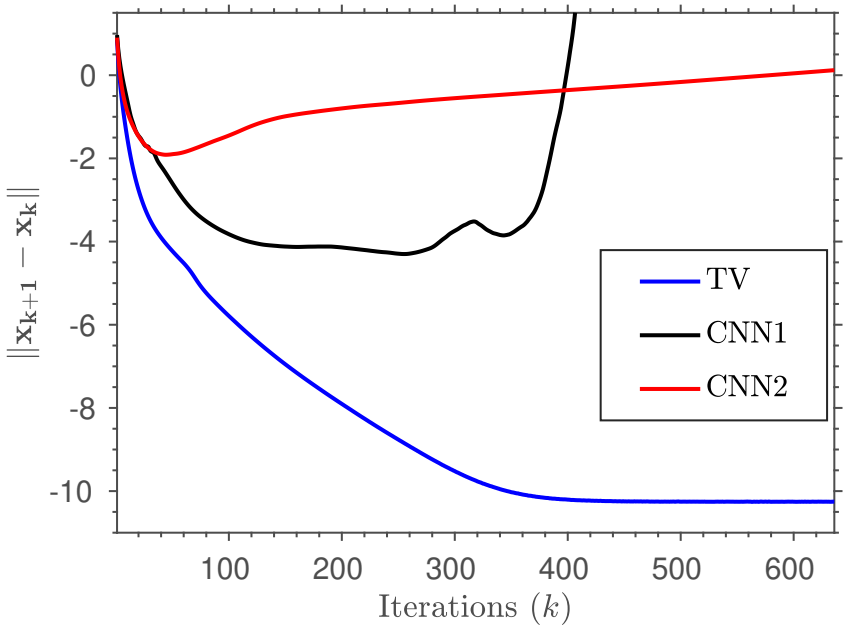


Figure 1.14: Divergence of $||\boldsymbol{x}_k-\boldsymbol{x}_{k-1}||_2$ for superresolution application using PnP-ISTA with two different CNN-based denoisers in [9] and [78]. The iterates converge in the case of PnP-ISTA with TV denoising.

We next compare PnP and RED algorithms with model-based optimization and end-to-end learning in Table 1.2. Note that in the case of end-of-end learning models, we consider their inference for comparison, since evaluation of models is indeed the algorithm for image

reconstruction. It is clear that PnP and RED have clear advantages over both model-based optimization algorithms and end-to-end learning-based methods. One disadvantage is that we need to tune the algorithm-specific parameters. However, a major issue with PnP and RED is that it is not known apriori if they converge. Let us explain this with an example. We can demonstrate many applications, where the PnP iterates $\boldsymbol{x}_k$ diverges and the algorithms return spurious estimates. We show two such applications in Table 1.3 and Fig. 1.14 for deblurring and superresolution. We can find similar examples for divergence of PnP-ADMM in [99] and divergence of RED in [97].

Table 1.2: Comparison of PnP and RED with optimization models and end-to-end-learning models.

Property	**Optimization models**	**End-to-End learning**	**PnP and RED**
Flexibility in handling various restoration tasks	Yes	No	Yes
Interpretability	Yes	No (CNNs) Yes (Deep Unfolding Networks)	Yes
Parameter tuning	Yes	No	Yes
Inference time	High	Low	Low
Performance	Low as compared to other two methods	High	High
Training data	No	Yes (excessive)	No

Table 1.3: Values of $||\boldsymbol{x}_k - \boldsymbol{x}_{k-1}||_2$ and PSNR (dB) for a debluring application using PnP-ISTA with DnCNN and total variation denoising. Since the former case does not have convergence guarantees, $||\boldsymbol{x}_k - \boldsymbol{x}_{k-1}||_2$ diverges resulting in spurious estimates (PSNR decreases drastically). In the latter case, we have convergence guarantees which are clear from the observations.

Denoiser		Number of iterations				
		10	25	50	75	100
DnCNN	$\|\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|\|_2$	146.00	146.58	234.55	2300	2207248
	PSNR	16.54	14.79	—4.09	—45.14	—84.68
TV	$\|\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|\|_2$	101.74	41.71	10.39	2.68	0.74
	PSNR	18.69	22.38	23.63	23.72	23.73

1.6 RED and PnP Convergence Theory

In general, PnP and RED lack convergence guarantees [97, 99] (also shown in counterexamples in Table 1.3 and Fig. 1.14) and hence can result in spurious reconstructions. This has resulted in wide interest in the convergence aspects of these algorithms. The question is what properties should the denoiser D satisfy to guarantee convergence of PnP and RED. Since this is at the heart of this thesis, we detail them next. We tabulate the convergence results for PnP algorithms in Table 1.4. The convergence results for RED are relatively few and we will discuss them later.

The problem of PnP convergence has been studied in several works. It was shown in [79] that a class of symmetric linear denoisers can be expressed as the proximal operator of a convex function, i.e., one can associate a convex regularizer g for every denoiser in this class, and the

Table 1.4: Summary of convergence results for PnP algorithms.

PAPER	Convergence	Comments	Condition on Denoiser	Parent Algorithm
Sreehari, 2016 [79]	Objective	Designed a linear denoiser with required conditions	Jacobian to be symmetric with eigenvalues in $[0, 1]$	ADMM
Chan, 2017 [100]	Iterate	Showed that NLM is not nonexpansive	bounded	ADMM-variant
Meinhardt, 2017 [101]		Proved equivalence of 4 algorithms with respect to convergence.		PG, ADMM, PDHG1, PDHG2
Dong , 2018 [45]	Iterate	Subsequence convergence results for linear inverse problems. The condition requires both denoiser & regularizer to be given.	a special descent condition	HQS
Teodoro , 2019 [88]	Objective	Derived explicit form of PnP regularizer.	Linear denoiser with same conditions as [79]	ADMM
Ryu, 2019 [78]	Iterate	Provided a heuristic algorithm to train a nonexpansive denoiser using power iteration	Residue of denoiser to be nonexpansive	FBS, DRS, ADMM
Sun, 2019 [81]	Iterate	Proved a weak form of convergence.	averaged	ISTA (stochastic)
Tirer and Giryes, 2019 [87]	Iterate	Provided an upper bound on the error of estimate for linear inverse problems.	bounded	IDBP
Chan, 2019 [102]		Proved several relations between PnP and graph Laplacian regularizers.		ADMM
Gavaskar, 2020 [103]	Iterate	Proved convergence for linear inverse problems	Linear, row-stochastic	ISTA
Yuan, 2020 [104]	Iterate	Proved convergence for linear inverse problems	Bounded	GAP (Generalized Alternating Projection)
Xu, 2020 [105]	Objective	Convergence to a stationary point of an objective function	MMSE estimator	ISTA
Sun, 2021 [82]	Iterate	Data-fidelity term f has to be separable	Residue of denoiser is 1/2-averaged	ADMM variant (stochastic)
Liu, 2021 [106]	Iterate	For compressed sensing	Denoiser residue is Lipschitz continuous, bounded	ISTA
Al-Shabili, 2022 [107]	Iterate	f has to be strongly convex	Lipschitz continuous	ISTA variant
Hurualt, 2022 [108]	Objective	Convergence to stationary points of a nonconvex functional	Gradient decent based denoiser	HQS variant
Cohen, 2022 [109]	Objective	Proposed denoisers has MAP and MMSE estimation	Gradient decent based denoiserd	Gradient descent variant
Hurualt, 2022 [110]	Objective	Denoiser corresponds to proximal operator of nonconvex function	Gradient decent based denoiser	ISTA, ADMM

PnP iterates in this case amount to minimizing $f+g$. The closed-form expression of g was later derived in [88]. For example, when the denoiser used is symmetric and linear, PnP-ISTA in (1.20) is equivalent to ISTA in (1.10) and hence convergence directly follows [40]. We refer to such type of convergence as "*objective convergence*". High-performance denoisers like BM3D [70] and trained denoisers like TNRD [71], DnCNN [9], etc. cannot be the proximal operator of a convex function. This is because the proximal map of a convex function must be nonexpansive and such denoisers have shown to violate nonexpansiveness [92]. Hence, most of the existing works for PnP algorithms address the question of convergence of updates, instead of optimality. We refer to this type of convergence results as "*iterate convergence*".

The iterate convergence results for PnP are proved for different combinations of the inverse problem, base algorithm, and denoiser in the literature. For example, iterate convergence was established for linear inverse problems with quadratic data-fidelity in [45, 87, 103]. In addition, the denoiser is assumed to satisfy a descent condition in [45], a boundedness condition in [87], and a linearity condition in [103]. On the other hand, the analysis in [12] applies to arbitrary convex data-fidelity but the denoiser is assumed to satisfy a boundedness condition in [12] (similar to [87]). PnP-ISTA convergence has been established, under mild assumptions, for generative denoisers for compressed sensing [20, 106], and GAN-based projectors for compressed sensing and phase retrieval [111]. A modification of PnP-ISTA is also shown to converge for Lipschitz denoisers and strongly convex f under more stringent assumptions [107]. Recently, motivated by RED, the authors in [108, 109, 110] proposed to construct denoisers by realizing them as a gradient descent step on a functional parameterized by a deep neural network. Such a denoiser (referred to as gradient descent-based denoiser) is used to prove iterate convergence of a variant of PnP-HQS, a variant of gradient descent, and PnP-ISTA and PnP-ADMM to stationary points of a nonconvex functional [108, 109, 110].

The works in [78, 81, 112, 113] reduce PNP-ISTA or PnP-ADMM to dynamical system of the form $\boldsymbol{x}_{k+1}=\mathrm{T}(\boldsymbol{x}_k)$ and understand the convergence of $\{\boldsymbol{x}_k\}$ using the properties of T. We require the following definitions to collect desirable properties of operator T which can effect convergence of PnP and RED.

Definition 1.1. *An operator* $\mathrm{T}:\mathbb{R}^n\to\mathbb{R}^n$ *is said to be,*

- β*-Lipschitz if* $\forall \boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^n:\ \|\mathrm{T}(\boldsymbol{x})-\mathrm{T}(\boldsymbol{y})\|\leqslant\beta\|\boldsymbol{x}-\boldsymbol{y}\|$.
- *nonexpansive if* $\beta=1$.
- *contractive if* $\beta\in[0,1)$.
- θ*-averaged for some* $\theta\in(0,1)$*, if* $\mathrm{T}=(1-\theta)\mathrm{I}+\theta\mathrm{T}_n$ *and* T_n *is nonexpansive.*

The authors in [78] showed that when $\mathrm{I}-\mathrm{D}$ is β-Lipschitz and f is strongly convex, then the iterations in PnP-ISTA (1.20) and PnP-ADMM in (1.21) can be written as $\boldsymbol{x}_{k+1} = \mathrm{T}(\boldsymbol{x}_k)$ for a contraction operator T, under some stringent assumptions on Lipschitz constant β. Then the authors use the Banach contraction principle to establish the convergence of the updates in (1.20) and (1.21). In particular, if $(\boldsymbol{x}_k)$ converges to a point $\boldsymbol{x}^*$, then by continuity of T, $\boldsymbol{x}^* = \mathrm{T}(\boldsymbol{x}^*)$. These points are called "fixed points" of T and this form of iterate convergence is also called fixed point convergence [114], since the limit point of the update $\boldsymbol{x}_k$ is a fixed point. Similarly, the averagedness property of D is used to prove a weaker form of convergence for PnP-ISTA and PnP-ADMM [81, 112] and demicontractivity property of denoiser is used to prove convergence of a variant of PnP-ISTA [113].

We next describe the available convergence results for RED. It has been empirically observed that RED can diverge if the denoiser is not nonexpansive (cf. Table II in [97]). In particular, RED converges if the denoiser has a symmetric Jacobian and is nonexpansive [91]. The authors in [97] developed a new block coordinate RED and showed that a nonexpansive denoiser is required for its convergence. In [115], RED is formulated as a convex optimization problem which includes applying a nonexpansive denoiser in the corresponding algorithm. In [106], the authors showed that a nonexpansive denoiser can also effect convergence of a variant of RED for compressed sensing. However, a new variant of RED is recently proposed in [98], where the denoiser need not be nonexpansive for algorithmic convergence. In summary, most of the existing convergence guarantees for RED [92, 97, 106, 115] require the denoiser D to be nonexpansive. Note that the relation between RED and PnP-ISTA is explored in [106, 115].

Our current understanding of the convergence of PnP and RED can be summarized as follows:

- for nonexpansive denoisers, convergence is guaranteed for RED [92, 97, 115], PnP-ISTA applied to compressive sensing [106], and a variant of PnP-ISTA [92].
- a weak form of convergence of PnP-ISTA and PnP-ADMM is guaranteed for averaged denoisers [81, 82].

For the above reason, there has been significant research on training averaged or more generally nonexpansive CNN denoisers. The core issue in this respect is that computing the Lipschitz constant of a CNN is intractable [116]. In particular, while the Lipschitz constant of a convolution layer is just the largest singular value of the matrix representation of the layer (which can be computed efficiently from the FFT of its filters [117]) and the Lipschitz constant of common nonlinear activations (ReLU and sigmoid) is 1, computing the Lipschitz constant for their cascaded composition is far from obvious. However, we can compute a (loose) upper bound for the Lipschitz constant, namely, the product of the largest singular value of the convolution layers [78]. Thus, for example, if we can force the singular value

of each convolutional layer to be within 1 at each epoch of the training process, then the final network is guaranteed to be nonexpansive. Several methods have been proposed in the literature that are essentially based on this idea, e.g., see [99, 117, 118, 119]. In particular, the authors in [78] constructed a nonexpansive CNN denoiser, where the largest singular value of a convolutional layer is estimated using power iterations, which is used to force the Lipschitz constant of the layer to be within 1 after each gradient step.

1.7 Open Problems

The key takeaways from the discussion on PnP and RED are as follows:

- Understanding convergence of PnP and RED is essential as there are pathological cases where the divergence of these algorithms results in spurious reconstructions.
- In the case of linear denoisers, convergence guarantees for PnP algorithms are available only for symmetric cases. But often high-performance linear denoisers such as non-local means are not symmetric.
- Training a nonexpansive CNN denoiser is important for the convergence of PnP and RED. However, a clear trade-off between denoising accuracy and nonexpansive property of a CNN denoiser was observed in [110]. In particular, the denoising performance of a provably nonexpansive CNN denoiser is off by at least 4-5 dB as compared to the best performing denoisers (which are not nonexpansive).

These observations show that there are multiple open problems in the application of PnP and RED for image reconstruction. Some of the open problems, which motivated this thesis, are listed next.

1) Does there exist a denoiser D that can effect the convergence of PnP or RED and demonstrate state-of-the-art regularization capabilities?

2) Are there any verifiable sufficient conditions for convergence of PnP and RED? Are these also necessary for convergence?

3) Given a denoiser D, what are the conditions for it to be proximable? In other words, does there exist a convex function g such that $\mathsf{D} = \mathrm{Prox}_g$?

4) Can PnP algorithms be seen as solving a convex problem $\min_{\boldsymbol{x}} h(\boldsymbol{x})$, when the denoiser is not symmetric unlike [79]? Is the problem solvable? Do PnP algorithms exhibit objective or iterate convergence? For example, do the updates $\boldsymbol{x}_k$ in (1.20) or (1.21) satisfy the condition $h(\boldsymbol{x}_k) \to h^*$, where h^* is the global minimum of h? Can we guarantee that $\boldsymbol{x}_k \to \boldsymbol{x}^*$ where $\boldsymbol{x}^*$ is such that $h(\boldsymbol{x}^*) = h^*$?

5) How close is the PnP or RED reconstruction to the ground-truth image? In other words, do there exist recovery guarantees for PnP and RED?

6) Since learning-based denoisers show state-of-the-art regularization capabilities, can we construct a trained denoiser which can effect PnP and RED convergence? How to go about training such a denoiser?

1.8 Contributions

In this thesis, we have tried to address some of the open problems in Section 1.7. It is already established in the literature that the denoisers being nonexpansive is sufficient for the convergence of RED. One of our main contributions is that we prove that the denoiser being averaged is sufficient to effect convergence of PnP-ISTA and PnP-ADMM, which has been one of the open problems. Note that our convergence results are stronger than [81, 112]. The idea is to show that when denoiser is averaged, PnP-ISTA and PnP-ADMM can be reduced to a dynamic system $\boldsymbol{x}_{k+1} = \mathrm{T}(\boldsymbol{x}_k)$, where T is averaged. Moreover, if T is averaged, we can prove that the updates $\boldsymbol{x}_k$ converge to a fixed point of T, provided the set of fixed points of T is nonempty.

Theorem (Informal). *If the data-fidelity term f is convex and ∇f is β-Lipschitz and* D *is θ-averaged for any $\theta \in (0,1)$, the iterates in PnP-ISTA converges. If the data-fidelity term f is convex and* D *is θ-averaged for any $\theta \in (0, 1/2]$, the iterates in PnP-ADMM converges.*

Our further contributions are explained next:

1. **PnP with linear denoisers:** Theoretical contributions include the following.

   - We establish both iterate and objective convergence for PnP algorithms for a special class of linear denoisers using our theoretical results on PnP algorithms with averaged denoisers.
   - Notably, unlike existing works where the focus is on symmetric denoisers, our analysis covers non-symmetric denoisers (thereby solving a long-standing open problem) such as nonlocal means and almost any convex data-fidelity. The novelty in this regard is that we work with a special inner product (and norm) derived from the linear denoiser; the latter requires us to appropriately define the gradient and proximal operators associated with the data fidelity term.
   - We characterize kernel denoisers as a proximal map of a convex function. We use this theoretical result to construct a quadratic regularizer. The novelty of the proposed quadratic regularizer is that, unlike classical quadratic regularizers, the quadratic form is derived from the observed data. One of our results tailored for nonlocal means denoiser is as follows.

Theorem (Regularizer for nonlocal means). *Let the nonlocal means denoiser be denoted as* $\mathrm{D}(\boldsymbol{x}) = \mathbf{D}^{-1}\mathbf{K}\boldsymbol{x}$. *Then* $\mathrm{D}(\boldsymbol{x})$ *is a proximal map of a convex regularizer.*

$$\mathrm{D}(\boldsymbol{x}) = \underset{\boldsymbol{x}}{\operatorname{argmin}} \quad \frac{1}{2}\boldsymbol{x}^\top \mathbf{D}(\mathbf{K}^{-1}\mathbf{D} - \mathbf{I})\boldsymbol{x} + \frac{1}{2}||\boldsymbol{x}||_{\mathrm{D}}^2,$$

where $||\boldsymbol{x}||_{\mathrm{D}} = \boldsymbol{x}^\top \mathbf{D}\boldsymbol{x}$.

Algorithmic contributions include the following.

- The quadratic form of the proposed regularizer reduces the optimization problem for linear inverse problems like superresolution, deblurring, inpainting, etc., to solving a linear system.
- We further adapt Krylov solvers for this linear system, where we obtain the reconstruction estimate in just a few computationally cheap iterations.
- The proposed algorithm is seen to match the learning-based regularization in terms of reconstruction quality. To the best of our knowledge, the possibility of achieving near state-of-the-art performance using a linear solver (quadratic regularizer) is never demonstrated before.

2. **PnP and RED with learning-based denoiser:** We show that nonlinear CNN-based denoisers do not satisfy the nonexpansive condition. In fact, learning nonexpansive denoisers with gradient-based learning is a challenging task—checking nonexpansivity is known to be computationally intractable. Using numerical examples, we show that existing CNN denoisers violate the nonexpansive property and can cause the PnP iterations to diverge. More precisely, algorithms for training nonexpansive denoisers either cannot guarantee nonexpansivity of the final denoiser or are computationally intensive. We propose to solve this open problem by moving away from CNN-based denoisers, learning nonexpansive (contractive) image denoisers by unfolding ISTA and ADMM iterations applied to wavelet denoising. In this regard, our proposal has the following advantages.

   - **Theoretical:** We train a provably contractive or averaged deep denoiser, which effects iterate convergence of PnP-ISTA, RED-PG [92] and algorithms such as [81, 82, 97, 115].
   - **Algorithmic:** We demonstrate that the regularization capacity of our trained denoisers for PnP and RED can be matched with CNN denoisers. To the best of our knowledge, this is the first work to propose a simple framework (computationally cheap training mechanism) for training provably averaged (contractive) denoisers using unfolding networks.

1.9 Organization

We have organized our work into three chapters.

- **Chapter 2:** We use of the theory of averaged operators for convergence analysis of PnP-ISTA and PnP-ADMM. In particular, we prove the existence of non-symmetric linear denoisers that are averaged with respect to a non-standard norm. We also construct a CNN-based denoiser which is empirically seen to be averaged with respect to the standard norm. We validate our theoretical findings and demonstrate the effectiveness of PnP algorithms. The chapter ends with a comparison of the proposed denoisers and possible areas of improvement.

- **Chapter 3:** The reconstruction returned by the PnP algorithm with a kernel denoiser is shown to be a minimizer of a convex objective function $f + g$. We then derive a fast algorithm (involving Krylov solvers) to solve the first-order optimality condition for $f + g$. We demonstrate empirical results to show that the proposed algorithm gets near to state-of-the-art performance for linear inverse problems.

- **Chapter 4:** We construct trained denoisers that are provably averaged (or contractive) by construction and hence automatically nonexpansive. We also demonstrate that the regularization capacity of constructed denoisers for PnP and RED can be matched with CNN denoisers

- **Chapter 5:** We summarize our contributions and discuss future research directions.

Chapter 2

Fixed-Point and Objective Convergence of PnP Algorithms

Abstract. *Although PnP regularization works surprisingly well in practice, the theoretical convergence of PnP iterations is less well understood and is currently an active research area. In this chapter, we first show that PnP convergence can be guaranteed using an averaged denoiser within the ISTA and ADMM frameworks for arbitrary (including non-quadratic and non-strongly) convex loss functions. Then, we establish both iterate and objective convergence for a special class of linear denoisers. Notably, unlike a few existing works where the focus is on symmetric denoisers, our analysis covers non-symmetric denoisers such as nonlocal means and almost any convex data-fidelity. The novelty in this regard is that we make use of our novel results on averaged operators and work with a special inner product (and norm) derived from the linear denoiser; the latter requires us to appropriately define the gradient and proximal operators associated with the data-fidelity term. So in effect, we show convergence for variants of standard PnP-ISTA and PnP-ADMM with linear denoisers. We then ask the question if we could construct a CNN-based denoiser that is averaged and thereby effect the convergence of standard PnP-ISTA and PnP-ADMM. We propose a method for the construction of such a deep denoiser which is empirically seen to be averaged. We validate the theoretical results and demonstrate the effectiveness of our proposals for image reconstruction problems.*

2.1 Introduction

In Chapter 1, we have seen why the convergence properties of PnP algorithms are important. In particular, a lack of convergence guarantees could result in spurious reconstructions. One such example for PnP-ISTA is shown in Table 1.3. The guarantees for convergence of PnP algorithms could be probed in two ways; convergence of PnP iterates (often referred to as fixed-point convergence) and whether the iterations minimize some objective function of the form $f + g$. Refer to Section 1.6 for a detailed description of the existing works on the convergence of PnP algorithms. In this chapter, the following key contributions are discussed:

1. We prove that convergence of the PnP iterates can be guaranteed for ISTA and ADMM provided f is convex and the denoiser is averaged (see Definition 2.1); f must be

differentiable for ISTA but is allowed to be non-smooth for ADMM. Our analysis is based on the theory of proximal and averaged operators [114]. In particular, we prove the existence of non-symmetric linear denoisers that are averaged with respect to a non-standard norm.

2. In Theorems 2.12 and 2.13, we simultaneously establish iterate and objective convergence for a special class of non-symmetric denoisers. In particular, this subsumes existing results on symmetric denoisers [79, 88]. Notably, unlike [78, 111, 120], our results hold for arbitrary convex data-fidelity. Our analysis highlights the need to work with a non-standard inner product derived from the denoiser. In effect, this requires us to appropriately define the gradient and proximal operators in ISTA and ADMM.

3. In Theorem 2.16, we prove that the NLM denoiser is the proximal operator of a quadratic convex function, where the norm in the proximal operator is induced by a non-standard inner product.

4. A CNN-based denoiser which is averaged w.r.t euclidean inner product is constructed using data transformation and projection method of spectral normalization. Though the constructed denoiser is only empirically seen to be averaged, it effects the convergence of PnP-ISTA and PnP-ADMM as shown in the experiments in Section 2.7. Moreover, we have also demonstrated pathological examples where the constructed denoiser is not averaged, which further led to the work in Chapter 4.

We validate our theoretical findings and demonstrate the effectiveness of the PnP algorithms using various image reconstruction applications. In particular, this chapter focuses on modified PnP-ISTA and PnP-ADMM with linear denoisers and standard PnP-ISTA and PnP-ADMM with the constructed CNN-based denoiser.

2.2 Preliminary Details

2.2.1 Notations

In this chapter and Chapter 3, we will work with non-standard inner products and norms on $\mathbb{R}^n$. In particular, we will require the notion of gradient, non-expansivity, proximal operator, etc. in a real Hilbert space $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$, where $\langle\cdot,\cdot\rangle$ is an abstract inner product on $\mathbb{R}^n$. We let $||\cdot||$ denote the norm induced by $\langle\cdot,\cdot\rangle$, i.e., $||\boldsymbol{x}|| = \langle \boldsymbol{x}, \boldsymbol{x}\rangle^{1/2}$ for $\boldsymbol{x} \in \mathbb{R}^n$. We will define the PnP iterations using the gradient and proximal operators corresponding to $\langle\cdot,\cdot\rangle$.

We denote the standard inner product on $\mathbb{R}^n$ by $\langle\cdot,\cdot\rangle_2$, i.e., $\langle \boldsymbol{x}, \boldsymbol{y}\rangle_2 = \boldsymbol{x}^\top \boldsymbol{y}$ for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. The norm induced by $\langle\cdot,\cdot\rangle_2$ is denoted by $||\cdot||_2$. We denote the identity operator on $\mathbb{R}^n$ by I, whereas the identity matrix is denoted by $\mathbf{I}$. The range space of a matrix $\mathbf{A}$ is denoted by $\mathscr{R}(\mathbf{A})$.

2.2.2 Basic Definitions

We begin by recalling the non-expansive and averaged operators on $\mathbb{R}^n$ from the previous chapter.

Definition 2.1. *An operator* T *on* $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ *is said to be non-expansive if* $||\mathrm{T}(\boldsymbol{x}) - \mathrm{T}(\boldsymbol{y})|| \leqslant ||\boldsymbol{x} - \boldsymbol{y}||$ *for all* $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. *An operator* G *on* $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ *is said to be* θ*-averaged, where* $\theta \in (0,1)$, *if we can write* $\mathrm{G} = (1-\theta)\mathrm{I} + \theta\mathrm{T}$, *where* T *is non-expansive. That is,* G *is* θ*-averaged if* $(1-1/\theta)\mathrm{I} + (1/\theta)\mathrm{G}$ *is non-expansive.*

We next define the gradient and the proximal operator in a Hilbert space $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$.

Definition 2.2. *A function* $f : \mathbb{R}^n \to \mathbb{R}$ *is said to be differentiable at* $\boldsymbol{x} \in \mathbb{R}^n$ *if there exists a (unique) linear map* $L : \mathbb{R}^n \to \mathbb{R}$ *(the derivative of* f *at* $\boldsymbol{x}$*) such that*

$$f(\boldsymbol{x} + \boldsymbol{h}) = f(\boldsymbol{x}) + L(\boldsymbol{h}) + o(||\boldsymbol{h}||) \qquad (\boldsymbol{h} \in \mathbb{R}^n), \tag{2.1}$$

where $||\cdot||$ *is any arbitrary norm on* $\mathbb{R}^n$. *If* (2.1) *holds at every* $\boldsymbol{x} \in \mathbb{R}^n$, *then we say that* f *is differentiable and we use* $L(\boldsymbol{x})$ *to denote the derivative at* $\boldsymbol{x}$. *For a fixed inner product* $\langle\cdot,\cdot\rangle$ *on* $\mathbb{R}^n$, *the corresponding gradient of* f *at* $\boldsymbol{x}$ *is the unique vector* $\nabla f(\boldsymbol{x}) \in \mathbb{R}^n$ *such that*

$$L(\boldsymbol{x})(\boldsymbol{h}) = \langle \nabla f(\boldsymbol{x}), \boldsymbol{h} \rangle \qquad (\boldsymbol{h} \in \mathbb{R}^n). \tag{2.2}$$

Note that the usual definition of ∇f as the vector of partial derivatives of f is consistent with Definition 2.2 when the inner product is $\langle\cdot,\cdot\rangle_2$ [121].

The proximal operator [114] is at the heart of algorithms such as ISTA and ADMM [38]. Let $\overline{\mathbb{R}}$ denote the extended real line $\mathbb{R} \cup \{\infty\}$. An extended-real-valued $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ is said to be proper if there exists $\boldsymbol{x} \in \mathbb{R}^n$ such that $f(\boldsymbol{x}) < \infty$. Moreover, f is said to be closed if its epigraph,

$$\mathrm{epi}(f) = \big\{(\boldsymbol{x}, t) \in \mathbb{R}^{n+1} : t \geqslant f(\boldsymbol{x})\big\},$$

is closed in $\mathbb{R}^{n+1}$.

Definition 2.3. *Let* $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ *be a closed, proper and convex function. The proximal operator* Prox_f *on* $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ *is defined as*

$$\mathrm{Prox}_f(\boldsymbol{x}) = \underset{\boldsymbol{y} \in \mathbb{R}^n}{\mathrm{argmin}} \ \frac{1}{2}||\boldsymbol{y} - \boldsymbol{x}||^2 + f(\boldsymbol{y}), \tag{2.3}$$

where $||\cdot||$ *is the norm induced by* $\langle\cdot,\cdot\rangle$.

2.2.3 Algorithms

The standard ISTA and ADMM algorithms for solving (1.6) in (1.10) and (1.12) can be generalized to $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ for any arbitrary $\langle\cdot,\cdot\rangle$, using the gradient and proximal operators in (2.2)

and (2.3). The ISTA iterations in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ are given by

$$\begin{aligned} z_{k+1} &= x_k - \rho^{-1}\nabla f(x_k). \\ x_{k+1} &= \mathrm{Prox}_{\rho^{-1}g}(z_{k+1}). \end{aligned}$$

where $\rho > 0$ and $x_0 \in \mathbb{R}^n$ is an arbitrary initialization. On the other hand, the ADMM iterations are given by

$$\begin{aligned} v_{k+1} &= \mathrm{Prox}_{\rho^{-1}f}(x_k - z_k), \\ x_{k+1} &= \mathrm{Prox}_{\rho^{-1}g}(v_{k+1} + z_k), \\ z_{k+1} &= z_k + v_{k+1} - x_{k+1}, \end{aligned}$$

where $\rho > 0$ and $x_0, z_0 \in \mathbb{R}^n$ are initializations. In the PnP variant of ISTA (ADMM), referred to as PnP-ISTA (PnP-ADMM), the proximal operator $\mathrm{Prox}_{\rho^{-1}g}$ is replaced by an arbitrary image denoiser D.

2.2.4 Image Reconstruction Applications

In this section, we brief about the preliminary details required to apply algorithms in Section 2.2.3 for different inverse problems. The details in this section will be referred to when we apply our proposed algorithm for image reconstruction in the upcoming chapters. In particular, we consider four image reconstruction applications: despeckling[5][122], superresolution[100, 123], deblurring[101, 123], and inpainting [87, 124]. As far as the algorithms PnP-ISTA and PnP-ADMM are concerned, the only quantity that depends on the specific reconstruction problem is the loss function f. The choice of f determines the z-update in PnP-ISTA and the v-update in PnP-ADMM (see (1.20) and (1.21) for a description of these updates). More specifically, we need to compute ∇f for the former and the proximal operator $\mathtt{Prox}_{\rho^{-1}f}$ for the latter. We provide these information for Euclidean space $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$ and record some relevant facts. We note that for superresolution, deblurring, and inpainting, the loss f is convex and ∇f is Lipschitz continuous [6, 7], thereby satisfying the assumptions for Theorems 2.8 and 2.9. Therefore, we apply both PnP-ISTA and PnP-ADMM for these applications.

Loss Functions

Despeckling. In M-look synthetic aperture radar imaging [122], the observation model is given by

$$s(i) = r_0(i)n(i), \tag{2.4}$$

where i is a given pixel in the image, $r_0 \in \mathbb{R}^n_{++}$ is the unknown reflectance image, and the components of $n \in \mathbb{R}^n_+$ are iid Gamma with unit mean and variance $1/M$. In particular, the

probability density of each component is

$$p_n(t) = \frac{M^M}{\Gamma(M)} t^{M-1} \exp(-Mt) \qquad (t \geqslant 0).$$

The above model is based on the assumption that the measurement $s(i)$ is an average of M independent samples of the intensity at pixel i [122].

Let $x_0(i) := \log r_0(i)$ and $y(i) := \log s(i)$. Taking logarithms on both sides of (2.4), we obtain the following additive model:

$$y(i) = x_0(i) + \log n(i),$$

where the transformed noise distribution is given by

$$p_{\log(n)}(t) = \frac{M^M}{\Gamma(M)} \exp\left(M(t - e^t)\right).$$

The loss function corresponding to the ML estimate of x_0 is given by [5]:

$$f(x) = M \sum_{\text{pixel } i} \Big[x(i) + \exp\big(y(i) - x(i)\big) \Big] + \text{constant}.$$

The gradient and Hessian of f w.r.t $\langle \cdot, \cdot \rangle_2$are:

$$[\nabla f(x)](i) = M\Big(1 - \exp\big(y(i) - x(i)\big)\Big).$$

and

$$[\nabla^2 f(x)](i,j) = \begin{cases} M \exp\big(y(i) - x(i)\big) & i = j, \\ 0 & i \neq j. \end{cases}$$

Clearly $\nabla^2 f(x) \succ 0$ for $x \in \mathbb{R}^n$, hence f is strictly convex. However, we cannot guarantee that $\mu \mathrm{I} \preceq \nabla^2 f(x) \preceq L\mathrm{I}$ for all $x \in \mathbb{R}^n$, where $\mu, L > 0$. Thus, f is not strongly convex and ∇f is not Lipschitz continuous in general. We will show in Theorem 2.8 that gradient of f being Lipschitz continuous is sufficient for convergence of PnP-ISTA. Note that f for despeckling violates this assumption.

On the other hand, we need the proximal map of f for PnP-ADMM which is given by:

$$\texttt{Prox}_{\rho^{-1} f}(z) = \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \; M \sum_{\text{pixel } i} \Big[x(i) + \exp\big(y(i) - x(i)\big) \Big] + \frac{\rho}{2} \sum_{\text{pixel } i} \big(x(i) - z(i)\big)^2. \tag{2.5}$$

The above optimization is separable, i.e., it can be decoupled into convex one-variable optimization problems. These in turn can be solved efficiently using Newton's method [5].

Superresolution. The observation model for image superresolution is given by:

$$y = \mathbf{S}\mathbf{B}x_0 + \eta, \tag{2.6}$$

where $x_0 \in \mathbb{R}^n$ is the unknown high-resolution image, $y \in \mathbb{R}^m$ is the observed low-resolution image ($m < n$), $\mathbf{B} \in \mathbb{R}^{n\times n}$ is a circulant matrix corresponding to a blur kernel b, and $\mathbf{S} \in \mathbb{R}^{m\times n}$ is a binary sampling matrix that decimates $\mathbf{B}x$ by factor K [100, 125]. For white Gaussian noise η, the loss function corresponding to the MAP estimate of x_0 is given by

$$f(x) = ||y - \mathbf{S}\mathbf{B}x||_2^2. \tag{2.7}$$

This is non-strongly convex since $\mathbf{S}$ has a non-trivial null space. The gradient of (2.7) is given by

$$\nabla f(x) = \mathbf{B}^\top \mathbf{S}^\top (\mathbf{S}\mathbf{B}x - y).$$

Note that $\mathbf{S}^\top \in \mathbb{R}^{n\times m}$ is an upsampling matrix and $\mathbf{B}^\top \in \mathbb{R}^{n\times n}$ is again a circulant matrix whose blur kernel is obtained by flipping b about the origin [100]. In particular, b is a symmetric Gaussian blur for our experiments and $\mathbf{B}^\top = \mathbf{B}$ in this case[1].

In this case, the proximal map is given by

$$\texttt{Prox}_{\rho^{-1}f}(z) = \underset{x\in\mathbb{R}^n}{\operatorname{argmin}} \; ||y - \mathbf{S}\mathbf{B}x||_2^2 + \frac{\rho}{2}\,||x - z||_2^2. \tag{2.8}$$

The minimizer can be computed in the closed form [126, Eqn. (15)].

Deblurring. Deblurring is a special case of superresolution where $K = 1$, $m = n$, and $\mathbf{S} = \mathbf{I}$. Thus, the loss function for deblurring is

$$f(x) = ||y - \mathbf{B}x||_2^2,$$

The gradient $\nabla f(x) = \mathbf{B}^\top(\mathbf{B}x - y)$ can be computed efficiently using FFT [100]. Moreover, the proximal map,

$$\texttt{Prox}_{\rho^{-1}f}(z) = \underset{x\in\mathbb{R}^n}{\operatorname{argmin}} \; ||y - \mathbf{B}x||_2^2 + \frac{\rho}{2}\,||x - z||_2^2.$$

can also be computed efficiently using FFT [100].

Inpainting. Image inpainting deals with estimating missing pixels (whose locations are

[1] In practice, $\mathbf{B}^\top$ or $\mathbf{B}$ are never populated and stored. They are applied by convolving the image with the corresponding blur kernel (using FFT for example).

known) in an image [87]. The observation model for inpainting is

$$y = \mathbf{S}x_0 + \eta,$$

where $x_0 \in \mathbb{R}^n$ is the ground-truth, $y \in \mathbb{R}^n$ is the observed image with 0 substituted for the missing pixels, and $\mathbf{S} \in \mathbb{R}^{n \times n}$ is a diagonal matrix given by

$$\mathbf{S}(i,i) = \begin{cases} 0 & \text{if pixel } i \text{ is missing,} \\ 1 & \text{otherwise.} \end{cases}$$

The loss function in this case is given by

$$f(x) = ||y - \mathbf{S}x||_2^2.$$

Since $\mathbf{S}$ has a non-trivial nullspace, f is non-strongly convex.

The required gradient and proximal maps are given by [38] :

$$\nabla f(x) = \mathbf{S}(\mathbf{S}x - y). \tag{2.9}$$

and

$$\texttt{Prox}_{\rho^{-1}f}(z) = (\mathbf{S} + \rho\mathbf{I})^{-1}(\mathbf{S}y + \rho z). \tag{2.10}$$

Since $\mathbf{S}$ is diagonal, we can compute (2.9) and (2.10) using pixelwise operations.

2.3 Fixed-point Convergence Results

In this section, we establish the following results for PnP-ISTA and PnP-ADMM:

- If the denoiser is averaged, then the iterates of PnP-ISTA and PnP-ADMM exhibit fixed-point convergence.
- The averaged property (with respect to a special inner product) is satisfied by a broad class of linear denoisers.
- For this class of linear denoisers, there exists a convex regularizer g such that the limit points of PnP-ISTA and PnP-ADMM are minimizers of $f + g$.

We will just state and discuss these results and connect them to existing results; their detailed derivations can be found in Section 2.10. As is well-known, the proximal operator (see Definition 2.3) of a closed, proper, and convex function is (1/2)-averaged in $(\mathbb{R}^n, \langle \cdot, \cdot \rangle_2)$ [38]. As a result, a denoiser that can be expressed as the proximal operator of a convex regularizer is averaged. For this reason, the symmetric linear denoisers in [79, 88] qualify as averaged

operators on $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$. But what about a generic linear denoiser $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$, where $\mathbf{W}$ has the basic properties of nonnegativity and row-stochasticity, but is possibly non-symmetric? The following result highlights that such denoisers do not qualify as averaged operators.

Proposition 2.4. *Let* D *be a linear operator on* $\mathbb{R}^n$. *In particular, let* $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$, *where* $\mathbf{W} \in \mathbb{R}^{n\times n}$.

1. *Suppose* $\mathbf{W}$ *is symmetric and its eigenvalues are in* $[0, 1]$. *Then* D *is* θ*-averaged on* $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$ *for all* $\theta \in [1/2, 1)$.

2. *Let* $\mathbf{W}$ *be nonnegative and row-stochastic (all rows sum to one), but not doubly-stochastic (some columns do not sum to one). Then* D *cannot be* θ*-averaged on* $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$ *for any* $\theta \in (0, 1)$.

In particular, Proposition 2.4 implies that kernel filters, such as those mentioned earlier [65, 66, 67, 68, 69] are not averaged on $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$. We remark that the weight matrix $\mathbf{W}$ is derived from the input image $\boldsymbol{x}$ in most of these filters. As a result, these filters are not strictly linear in terms of the input-output relation. However, $\mathbf{W}$ can be computed from a surrogate image [127] or fixed after a few PnP iterations [79, 102, 103, 128, 129, 130, 131]. The filter can be treated as a linear operator $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$ in this case [132]. In particular, while $\mathbf{W}$ is nonnegative and row-stochastic, it is naturally non-symmetric. Hence, D cannot possibly be θ-averaged on $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$.

The above negative result leads us to the natural question: Is $\mathbf{W}$ averaged with respect to some non-standard inner product? We show in Proposition 2.15 that this is indeed the case. In particular, we will fix an appropriate inner product $\langle\cdot,\cdot\rangle$ on $\mathbb{R}^n$ and use the corresponding gradient and proximal operators within PnP-ISTA and PnP-ADMM. More precisely, we consider the following PnP-ISTA iterations:

$$\boldsymbol{x}_{k+1} = \mathrm{D}(\boldsymbol{x}_k - \rho^{-1}\nabla f(\boldsymbol{x}_k)), \tag{2.11}$$

and the following PnP-ADMM iterations:

$$\boldsymbol{v}_{k+1} = \mathrm{Prox}_{\rho^{-1}f}(\boldsymbol{x}_k - \boldsymbol{z}_k), \tag{2.12a}$$

$$\boldsymbol{x}_{k+1} = \mathrm{D}(\boldsymbol{v}_{k+1} + \boldsymbol{z}_k), \tag{2.12b}$$

$$\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \boldsymbol{v}_{k+1} - \boldsymbol{x}_{k+1}. \tag{2.12c}$$

With the averaged property in place for kernel filters, we can establish convergence of PnP-ISTA and PnP-ADMM using such filters. Notably, we do not need to symmetrize $\mathbf{W}$ that entails additional cost [79].

We now analyze the fixed-point convergence of PnP-ISTA and PnP-ADMM for averaged denoisers $\mathrm{D} : \mathbb{R}^n \to \mathbb{R}^n$. This is based on the fixed-point theory of averaged operators [114].

Definition 2.5. *We say that* $\boldsymbol{x}^* \in \mathbb{R}^n$ *is a fixed point of* $\mathrm{T} : \mathbb{R}^n \to \mathbb{R}^n$ *if* $\mathrm{T}(\boldsymbol{x}^*) = \boldsymbol{x}^*$. *The set of fixed points of* T *is denoted by* $\mathrm{fix}(\mathrm{T})$.

We state a lemma [114] that is required to establish convergence of PnP-ISTA and PnP-ADMM.

Lemma 2.6. *Let* $\mathrm{T} : \mathbb{R}^n \to \mathbb{R}^n$ *be* θ*-averaged on* $(\mathbb{R}^n, \langle \cdot, \cdot \rangle)$, *where* $\theta \in (0,1)$. *Assume that* $\mathrm{fix}(\mathrm{T})$ *is not empty and let* $\boldsymbol{x}_0 \in \mathbb{R}^n$. *Then the sequence* $(\boldsymbol{x}_k)_{k \geqslant 0}$ *generated as* $\boldsymbol{x}_{k+1} = \mathrm{T}(\boldsymbol{x}_k)$ *converges to some* $\boldsymbol{x}^* \in \mathrm{fix}(\mathrm{T})$.

To apply this result to PnP-ISTA, we need the notion of a smooth function.

Definition 2.7. *Let* $f : \mathbb{R}^n \to \mathbb{R}$ *be differentiable. It is said to be* β*-smooth on* $(\mathbb{R}^n, \langle \cdot, \cdot \rangle)$ *if there exists* $\beta > 0$ *such that* $||\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})|| \leqslant \beta ||\boldsymbol{x} - \boldsymbol{y}||$ *for all* $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, *where* $||\cdot||$ *is the norm induced by* $\langle \cdot, \cdot \rangle$.

We are now ready to state our main results on the fixed-point convergence of PnP-ISTA and PnP-ADMM. Henceforth, we assume that the data-fidelity term $f : \mathbb{R}^n \to \mathbb{R}$ is real-valued (rather than extended real-valued), which is the case with most imaging applications.

Theorem 2.8. *Let* $\mathrm{T}_{\mathrm{ISTA}} = \mathrm{D} \circ (\mathrm{I} - \rho^{-1} \nabla f)$. *Suppose that*

- f *is convex and* β*-smooth,*
- D *is* θ*-averaged for some* $\theta \in (0,1)$, *and*
- $\mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}}) \neq \varnothing$.

Then for any $\boldsymbol{x}_0 \in \mathbb{R}^n$ *and* $\rho > \beta/2$, *the sequence* $(\boldsymbol{x}_k)_{k \geqslant 0}$ *generated by* (2.11) *converges to some* $\boldsymbol{x}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}})$.

We remark that in a recent work [113], among other things, fixed-point convergence of PnP-ISTA was shown for denoisers that are derived from *demicontractive* operators and their averages. We note that our settings for convergence of PnP-ISTA include denoisers obtained by averaging nonexpansive operators, which form a subclass of demicontractive operators. However, the technical assumptions required for the theoretical results in [113] are different from those in Theorem 2.8; hence Theorem 2.8 and the convergence results for PnP-ISTA in [113] do not subsume each other.

Theorem 2.9. *Define*

$$\mathrm{T}_{\mathrm{ADMM}} = \frac{1}{2}\mathrm{I} + \frac{1}{2}\left(2\mathrm{Prox}_{\rho^{-1} f} - \mathrm{I}\right) \circ (2\mathrm{D} - \mathrm{I}). \tag{2.13}$$

Suppose that

- *f is convex,*
- D *is θ averaged for some $\theta \in (0, 1/2]$, and*
- $\mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}}) \neq \emptyset$.

Then, for arbitrary $\rho > 0$ and $\boldsymbol{x}_0, \boldsymbol{z}_0 \in \mathbb{R}^n$, the sequence $(\boldsymbol{v}_k, \boldsymbol{x}_k, \boldsymbol{z}_k)_{k \geqslant 1}$ generated by (2.12) *is convergent and the limit point is determined by some $\boldsymbol{u}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})$.*

Note that the results in Theorems 2.8 and 2.9 hold for any choice of the inner product $\langle \cdot, \cdot \rangle$. Moreover, D is not assumed to be linear. However, it is assumed that $\mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}})$ and $\mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})$ are non-empty. There are a couple of issues in this regard. First, verifying whether a given denoiser is averaged is not an easy task; this is especially true for non-linear denoisers such as BM3D [70] and DnCNN [9]. Second, even if the denoiser D is averaged, it is unclear whether $\mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}})$ and $\mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})$ are non-empty. In many cases, the latter condition is not verifiable and is simply assumed to hold without proof [81]. In the subsequent discussion, we show how to deal with these issues for a special class of linear denoisers and how to construct a CNN-based denoiser that seems to be empirically (not provably) averaged.

2.4 Linear Denoisers

In this section, we deal with a special class of linear denoisers, which lead to convergent PnP algorithms.

Definition 2.10. *Let $\mathfrak{L}$ denote the class of linear denoisers $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$ on $\mathbb{R}^n$ such that $\mathbf{W}$ is diagonalizable and its eigenvalues are in $[0, 1]$.*

In the following theorem, we collect some relevant properties of $\mathfrak{L}$, particularly that every denoiser in $\mathfrak{L}$ is averaged.

Theorem 2.11. *Let D be in class $\mathfrak{L}$ and $\mathbf{W}$ be the associated matrix, i.e., $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$. Let $\mathbf{V} \in \mathbb{R}^{n \times n}$ be an eigen matrix of $\mathbf{W}$, i.e., the columns of $\mathbf{V}$ are linearly independent eigenvectors of $\mathbf{W}$. Define the inner product*

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \left\langle \mathbf{V}^{-1}\boldsymbol{x}, \mathbf{V}^{-1}\boldsymbol{y} \right\rangle_2. \tag{2.14}$$

Then we have the following properties:

1. D *is the proximal operator on $(\mathbb{R}^n, \langle \cdot, \cdot \rangle)$ of some closed, proper and convex function $h_{\mathrm{D}} : \mathbb{R}^n \to \overline{\mathbb{R}}$.*
2. D *is θ-averaged on $(\mathbb{R}^n, \langle \cdot, \cdot \rangle)$ for every $\theta \in [1/2, 1)$.*
3. *The restriction of h_{D} to $\mathscr{R}(\mathbf{W})$ is real-valued (and hence continuous).*

4. *The iterates of PnP-ISTA in* (2.11) *(resp. PnP-ADMM in* (2.12)*) are identical to that of ISTA (resp. ADMM) applied to the optimization problem*

$$\min_{x\in\mathbb{R}^n} f(x)+\rho h_{\mathrm{D}}(x). \tag{2.15}$$

Since (2.14) depends on D via an eigenbasis of $\mathbf{W}$, we will refer to this as an inner product induced by D. We can interpret (2.14) as the standard inner product applied along with a change of basis, i.e., we perform our computations with respect to an eigenbasis of $\mathbf{W}$ instead of the standard basis. In particular, if $\mathbf{W}$ is symmetric, then (2.14) is in fact $\langle\cdot,\cdot\rangle_2$ since the eigen matrix $\mathbf{V}$ can be taken to be orthogonal in this case.

If the denoiser D is in class $\mathfrak{L}$ and $\langle\cdot,\cdot\rangle$ is an inner product induced by D, then the iterates in PnP-ISTA and PnP-ADMM are guaranteed to converge to a minimizer of (2.15).

Theorem 2.12. *Let* $\mathrm{D}\in\mathfrak{L}$ *and* $\langle\cdot,\cdot\rangle$ *be an inner product induced by* D. *Consider the space* $(\mathbb{R}^n,\langle\cdot,\cdot\rangle)$. *Assume that*

- f *is convex and* β*-smooth.*
- x^* *is a minimizer of* (2.15) *and* $p^*=f(x^*)+\rho h_{\mathrm{D}}(x^*)$.

Then, for any $x_0\in\mathbb{R}^n$ *and* $\rho>\beta/2$,

1. *the PnP-ISTA iterates* $(x_k)_{k\geqslant 0}$ *generated by* (2.11) *converge to a minimizer of* (2.15), *and*
2. $\lim_{k\to\infty} f(x_k)+\rho h_{\mathrm{D}}(x_k)=p^*$.

Theorem 2.13. *Let* $\mathrm{D}\in\mathfrak{L}$ *and* $\langle\cdot,\cdot\rangle$ *be an inner product induced by* D. *Consider the space* $(\mathbb{R}^n,\langle\cdot,\cdot\rangle)$. *Assume that* f *is convex,* y^* *is a minimizer of* (2.15) *and* $p^*=f(y^*)+\rho h_{\mathrm{D}}(y^*)$. *Then, for any* $\rho>0$ *and initialization* $x_0,z_0\in\mathbb{R}^n$,

1. *the PnP-ADMM iterates* $(x_k)_{k\geqslant 0}$ *generated by* (2.12b) *converge to a minimizer of* (2.15), *and*
2. $\lim_{k\to\infty} f(x_k)+\rho h_{\mathrm{D}}(x_k)=p^*$.

We note that Theorem 2.13 subsumes the objective convergence result in [79], where $\mathbf{W}$ is assumed to be symmetric. The above theorems show that convergence of PnP algorithms can be extended to a larger class of linear denoisers $\mathfrak{L}$ including non-symmetric denoisers, provided we work with a special inner product. The practical utility of the convergence results is that many kernel filters belong to the class $\mathfrak{L}$. Though this is well-known [132], we explain why this is so for completeness. Let $X:\Omega\to\mathbb{R}$ be the input image. The elements of X are

$\{X_s : s \in \Omega \subset \mathbb{Z}^2\}$, where Ω is the support of the image, $s \in \Omega$ is a pixel location, and X_s is the corresponding intensity value. Let $\zeta_s \in \mathbb{R}^M$ represent some feature at pixel s. The output of a generic kernel filter is given by

$$(\mathfrak{D}(X))_s = \frac{\sum_{t\in\Omega} \phi(\zeta_s, \zeta_t) X_t}{\sum_{t\in\Omega} \phi(\zeta_s, \zeta_t)}, \tag{2.16}$$

where the kernel function $\phi : \mathbb{R}^M \times \mathbb{R}^M \to \mathbb{R}$ is nonnegative and symmetric [133]. It is clear from (2.16) that $\mathfrak{D}$ is a linear operator. To obtain the matrix representation of (2.16), we need to linearly index the pixel space Ω. Let $\sigma : [1, N] \to \Omega$, where N is the number of pixels, be such that every index $\ell \in [1, N]$ is mapped to a unique pixel $\sigma(\ell) \in \Omega$. Let $\boldsymbol{x}$ be the vector representation of image X corresponding to this indexing, i.e., $\boldsymbol{x}_\ell = X_{\sigma(\ell)}$ for $\ell \in [1, N]$. Now, define the kernel matrix $\mathbf{K} \in \mathbb{R}^{N\times N}$ to be

$$\forall\ i, j \in [1, N]: \quad \mathbf{K}_{i,j} = \phi(\zeta_{\sigma(i)}, \zeta_{\sigma(j)}), \tag{2.17}$$

and the diagonal matrix $\mathbf{D} \in \mathbb{R}^{n\times n}$ as

$$\mathbf{D}_{i,i} = \sum_{j=1}^{N} \mathbf{K}_{i,j}. \tag{2.18}$$

for $i \in [1, N]$. We can then express (2.16) as the linear transform $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$, where

$$\mathbf{W} := \mathbf{D}^{-1}\mathbf{K}. \tag{2.19}$$

Note that D in (2.16) is linear in $\boldsymbol{x}$ if $\mathbf{K}$ does not depend on $\boldsymbol{x}$.

Definition 2.14. *A kernel function $\phi : \mathbb{R}^M \times \mathbb{R}^M \to \mathbb{R}$ is said to be positive definite if for any $N \geqslant 1$, $c_1, \dots c_N \in \mathbb{R}$, and $\boldsymbol{x}_1, \dots \boldsymbol{x}_N \in \mathbb{R}^M$,*

$$\sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j \phi(\boldsymbol{x}_i, \boldsymbol{x}_j) \geqslant 0.$$

Proposition 2.15. *If the kernel ϕ in* (2.16) *is positive definite and* D *is given by* (2.19)*, then* D *belongs to $\mathfrak{L}$. Furthermore,* D *is θ-averaged on $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_{\mathbf{D}})$ for every $\theta \in [1/2, 1)$, where $\langle\cdot,\cdot\rangle_{\mathbf{D}}$ is given by*

$$\langle \boldsymbol{x}, \boldsymbol{y}\rangle_{\mathbf{D}} = \boldsymbol{x}^\top \mathbf{D}\boldsymbol{y},$$

with $\mathbf{D}$ *as in* (2.18).

For more information on kernel filters, we refer the reader to [133, 134]. In our experiments, we use the nonlocal means (NLM) denoiser, which is a special instance of (2.16). In

NLM, the feature vector is given by $\zeta_s = (s, \mathbf{P}_s)$, where $s \in \mathbb{R}^2$ is the pixel coordinates and $\mathbf{P}_s$ is a (vectorized) image patch around pixel s of a guide image. We note that the guide image is a fixed image and is not, in general, equal to the image to be denoised. This makes the denoiser linear in its input. In particular, we consider the following NLM kernel:

$$\phi(\zeta_s, \zeta_t) = \Lambda(s-t)\mu(\mathbf{P}_s - \mathbf{P}_t), \tag{2.20}$$

where μ is Gaussian and the hat function $\Lambda : \mathbb{R}^2 \to \mathbb{R}$ is given by

$$\Lambda(s) = \prod_{i=1}^{2} \left(1 - \left|\frac{s_i}{N_s}\right|\right)_+,$$

where $(t)_+ = \max(0, t)$ and N_s is the search radius. The kernel in (2.20) is positive definite [79]. Importantly, if we define $\mathbf{K}$ and $\mathbf{D}$ as in (2.17) and (2.18) and the NLM denoiser using (2.19) with a fixed guide image (used to compute the kernel in (2.20)), then the denoiser belongs to $\mathfrak{L}$. Furthermore, the denoiser is the proximal operator of a quadratic convex regularizer.

Theorem 2.16. *The kernel matrix* $\mathbf{K}$ *in NLM is positive definite. Furthermore the corresponding denoiser* (2.19) *is the proximal operator on* $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ *of the convex regularizer*

$$h_{\mathrm{D}}(x) = \frac{1}{2} x^\top \mathbf{D}(\mathbf{K}^{-1}\mathbf{D} - \mathbf{I})x, \tag{2.21}$$

where $\langle\cdot,\cdot\rangle$ *is as specified in Proposition 2.15.*

We conclude this section with a discussion of the scope of our analysis. Our proofs are intricately tied to the form of the updates in ISTA and ADMM; it is not clear whether they can be adapted to other PnP algorithms. In particular, our analysis cannot directly be applied to accelerated variants such as PnP-FISTA [81]. We also note that the objective convergence results (Theorems 2.12 and 2.13) are only applicable to linear denoisers, and in practice, kernel filters. In particular, for a kernel filter to be treated as a linear denoiser, we are required to fix $\mathbf{W}$ after a finite number of PnP iterations; that is, we cannot use the reconstruction to adapt $\mathbf{W}$ beyond this point.

2.5 Numerical Results for Linear Denoisers

In this section, we validate the convergence results for PnP-ISTA and PnP-ADMM established in Section 2.4 using a couple of image reconstruction experiments—superresolution and despeckling (our source code is available at [135]). Recall that the data-fidelity f is quadratic for the former and non-quadratic for the latter. Importantly, f is convex but not strongly convex. Thus, we are able to numerically validate our results under very general conditions. We chose these applications as both have non-strongly convex data-fidelity terms f, with f

Table 2.1: PSNR/SSIM values for superresolution of Set12 dataset [9] using PnP-ISTA.The σ values are on a scale of 0 to 255.

Settings	Method	PnP in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$	PnP in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$
$K=2$	$\sigma=5$	28.15/0.830	28.20/0.836
	$\sigma=10$	26.80/0.775	26.94/0.785
$K=4$	$\sigma=5$	24.31/0.737	24.29/0.726
	$\sigma=10$	23.38/0.689	23.43/0.692

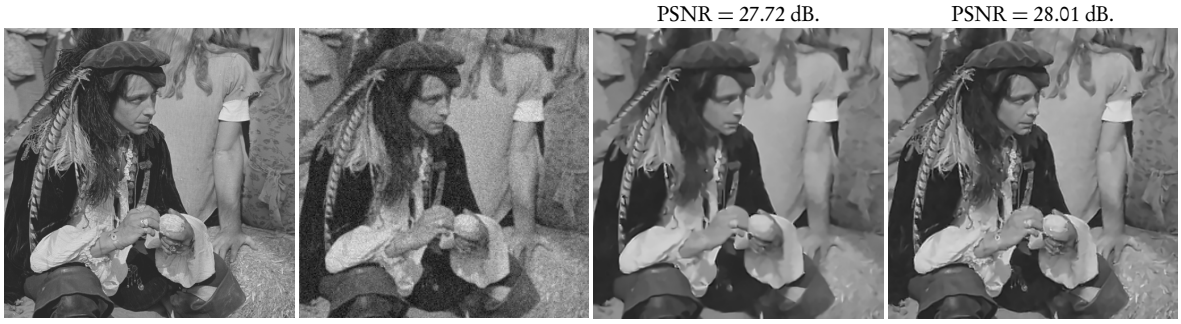


Figure 2.1: Image superresolution for $K=2$ and $\sigma=10$ using PnP-ISTA. The observed image has size 256×256 (scaled for display purposes), whereas the other images have size 512×512. The third and fourth images are the reconstructions from PnP-ISTA in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$ and $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$. The respective SSIM values are 0.728 and 0.747, while the PSNR values are reported above the images.

being quadratic for superresolution and non-quadratic for despeckling. We use NLM as the denoiser (with the kernel in (2.20)) and we work in the space defined by the inner product $\langle\cdot,\cdot\rangle$ in Proposition 2.15. The matrix $\mathbf{D}$ is computed from the image obtained after five PnP iterations, and the weight matrix $\mathbf{W}$ is kept fixed thereafter. Thus, the linear denoiser $\mathrm{D}(\boldsymbol{x})=\mathbf{W}\boldsymbol{x}$ belongs to $\mathfrak{L}$. The purpose of the experiments is solely to demonstrate that iterate and objective convergence are indeed achieved in practical imaging problems, as predicted by Theorems 2.12 and 2.13. In particular, we do not claim that our reconstructions are superior to existing methods, including PnP with other denoisers. Nevertheless, since PnP algorithms in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ (for non-standard inner products) have not been used till date, we compare the reconstruction quality with PnP in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$ (i.e. standard PnP). This is done to confirm that a similar reconstruction quality is obtained regardless of the inner product used. We stress that in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$, convergence guarantees for PnP-ISTA and PnP-ADMM with NLM denoiser are not available in full generality. Thus, working with the appropriate (denoiser-induced) inner product offers the advantage of a better understanding of the PnP mechanism (via an objective function), in addition to establishing convergence.

Note that in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$, the expression for ∇f in PnP-ISTA and $\mathrm{Prox}_{\rho^{-1}f}$ in PnP-ADMM contains $\mathbf{D}$; also see (2.22) and (2.23) later in this section. These expressions are different from those given in Section 2.2.4. In this sense, a knowledge of the denoiser is necessary even for

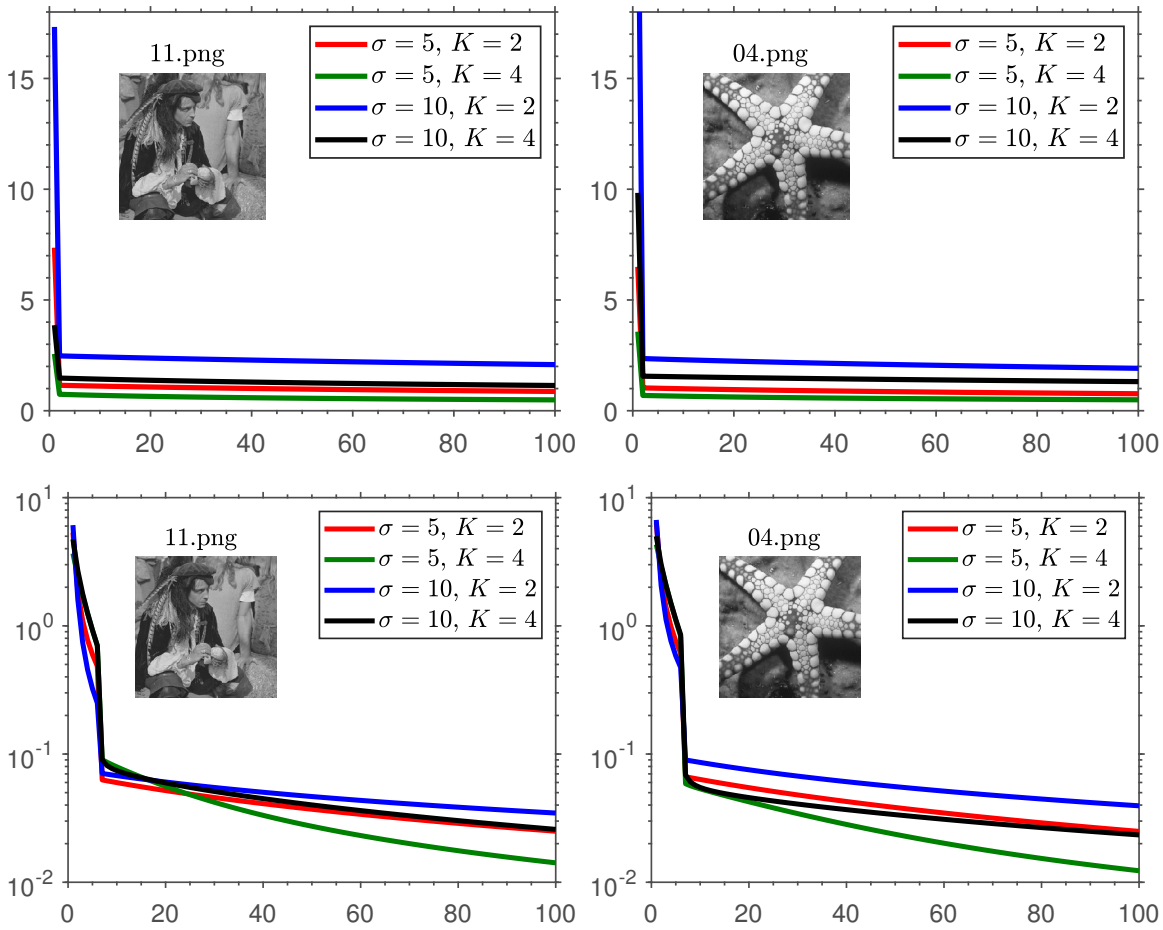


Figure 2.2: Objective and residual convergence for PnP-ISTA in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ for $\rho = 2.5$. Top row: $f(\boldsymbol{x}_k) + \rho h_{\mathrm{D}}(\boldsymbol{x}_k)$. Bottom row: $\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2$ (on a logarithmic scale).

computing quantities that depend on the data-fidelity f. Thus, PnP algorithms in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ involve more than just applying the denoiser in the denoising step. This is in contrast to PnP algorithms operating in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$.

Superresolution

Refer to (2.6) for details about the forward model for superresolution application. We use PnP-ISTA to estimate the ground-truth high-resolution image. The gradient of the data-fidelity term for superresolution in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_{\mathbf{D}})$ where $\langle\cdot,\cdot\rangle_{\mathbf{D}}$ is as specified in Proposition 2.15, is given by

$$\nabla f(\boldsymbol{x}) = \mathbf{D}^{-1}\mathbf{B}^{\top}\mathbf{S}^{\top}(\mathbf{S}\mathbf{B}\boldsymbol{x} - \boldsymbol{y}). \tag{2.22}$$

In our experiments, b is a symmetric Gaussian blur for our experiments and $\mathbf{B}^{\top} = \mathbf{B}$ in this case. In practice, $\mathbf{B}^{\top}$ and $\mathbf{B}$ are never populated and stored. They are applied using FFT-based convolution. For all experiments, we use a symmetric Gaussian blur of size 9×9 and standard deviation 1.

In Table 2.1, we report PSNR/SSIM values, averaged over the images (resized to 256×256)

Table 2.2: Objective and iterate convergence of PnP-ADMM in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ for image despeckling ($\rho = 0.2$).

Image	Settings	$f(x_k)+\rho h_{\mathrm{D}}(x_k)-C$						$\|x_{k+1}-x_k\|_2$				
		$k=1$	$k=10$	$k=20$	$k=30$	$k=40$	C	$k=1$	$k=10$	$k=20$	$k=30$	$k=40$
03.png	$M=5$	1.8×10^{3}	2.7	1.6×10^{-2}	1.9×10^{-4}	3×10^{-6}	3.4×10^{6}	7.5×10^{1}	2.6×10^{-1}	1.3×10^{-2}	1.2×10^{-3}	1.4×10^{-4}
	$M=7$	2.7×10^{3}	8.4	1.1×10^{-1}	2.6×10^{-3}	8.9×10^{-5}	4.7×10^{6}	6.7×10^{1}	3.2×10^{-1}	2.4×10^{-2}	3.1×10^{-3}	5.1×10^{-4}
	$M=10$	4.1×10^{3}	2.4×10^{1}	5.9×10^{-1}	2.6×10^{-2}	1.7×10^{-3}	6.8×10^{6}	6.3×10^{1}	3.9×10^{-1}	4.2×10^{-2}	7.3×10^{-3}	1.6×10^{-3}
05.png	$M=5$	2.8×10^{3}	3.9	1.8×10^{-2}	1.6×10^{-4}	2.0×10^{-6}	3.3×10^{6}	8.0×10^{1}	3.1×10^{-1}	1.4×10^{-2}	1.1×10^{-3}	1.1×10^{-4}
	$M=7$	4.1×10^{3}	1.2×10^{1}	1.2×10^{-1}	2.1×10^{-3}	5.1×10^{-5}	4.7×10^{6}	7.1×10^{1}	3.9×10^{-1}	2.7×10^{-2}	3.0×10^{-3}	4.1×10^{-4}
	$M=10$	6.2×10^{3}	3.5×10^{1}	6.6×10^{-1}	2.0×10^{-2}	8.2×10^{-4}	6.7×10^{6}	6.6×10^{1}	4.9×10^{-1}	4.8×10^{-2}	7.1×10^{-3}	1.3×10^{-3}

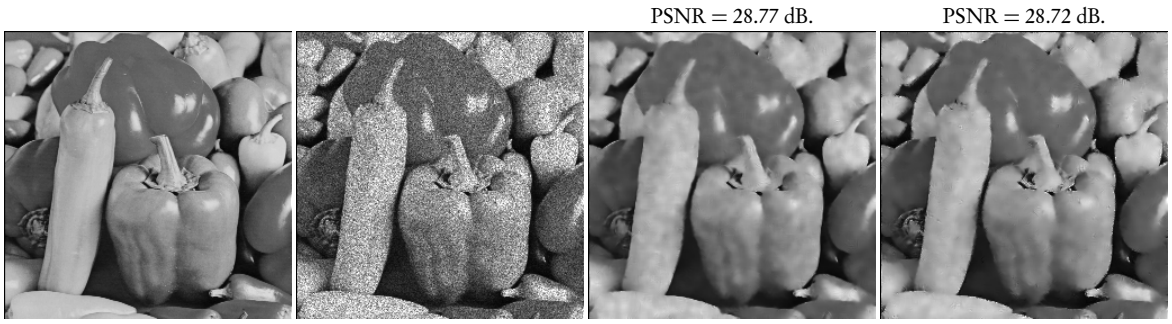


Figure 2.3: Image despeckling for $M = 5$ using PnP-ADMM. The third and fourth images are the reconstructions from PnP-ADMM in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$ and $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$. The respective SSIM values are 0.840 and 0.847, while the PSNR values are reported above the images.

in the Set12 dataset [9]. Note that the reconstruction quality using PnP-ISTA in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ is competitive with its standard counterpart. In particular, we note that this behavior holds for different values of sampling factor K and σ. For completeness, a visual result is shown in Fig. 2.1. We empirically verify Theorem 2.12 for two images from the Set12 dataset and using different values of K and σ; the results are reported in Fig. 2.2. Notice that the objective value corresponding to the proposed algorithm decreases in every iteration and stabilizes. Such behaviour cannot be verified for the standard counterpart since the existence of an objective function is not known. As for the iterates (x_k), it is not possible to directly verify the convergence since the limit point x^* is not known. Instead, as shown in the figure, we verify a necessary condition, namely that $\|x_k - x_{k-1}\|_2$ decays to 0 as k increases. Also, notice that the successive difference of iterates $\|x_k - x_{k-1}\|_2$ decay to 0; this is necessary for iterate convergence but not sufficient though.

Despeckling

Refer to (2.4) for details about the forward model for despeckling application. We use PnP-ADMM to estimate the ground-truth image. Note that the proximal operator of f in the space $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_{\mathbf{D}})$ is given by:

$$\operatorname{Prox}_{\rho^{-1}f}(u) = \operatorname*{argmin}_{x\in\mathbb{R}^n} \left\{ M \sum_i \Big[x(i) + \exp\big(y(i) - x(i)\big) \Big] + \frac{\rho}{2}(x-u)^{\top}\mathbf{D}(x-u) \right\}. \tag{2.23}$$

Table 2.3: PSNR/SSIM values for despeckling of Set12 dataset [9] using PnP-ADMM.

Settings \ Method	PnP in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$	PnP in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$
$M = 5$	27.65/0.809	27.49/0.793
$M = 10$	29.37/0.838	29.01/0.829

The above optimization is separable since $\mathbf{D}$ is diagonal. In fact, the optimization can be decoupled into one-variable convex problems, which can be solved efficiently using Newton's method [5].

A visual example of despeckling of a simulated observation is shown in Fig. 2.3 for $M = 5$. The values of $f(\boldsymbol{x}_k) + \rho h_{\mathrm{D}}(\boldsymbol{x}_k)$ and $\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2$ for different values of M and input images are reported in Table 2.2. It is evident that $f(\boldsymbol{x}_k) + \rho h_{\mathrm{D}}(\boldsymbol{x}_k)$ converges to a stable value, whereas $\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2$ decays to 0 with iterations k. These observations agree with Theorem 2.13. We compare the reconstruction quality with PnP-ADMM in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$ by averaging the PSNR and SSIM values over the images in the Set12 dataset (all images were resized to 256×256). These are reported in Table 2.3. Note that the PSNR/SSIM values are comparable for PnP-ADMM in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ and $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$.

2.6 Averaged CNN Denoiser

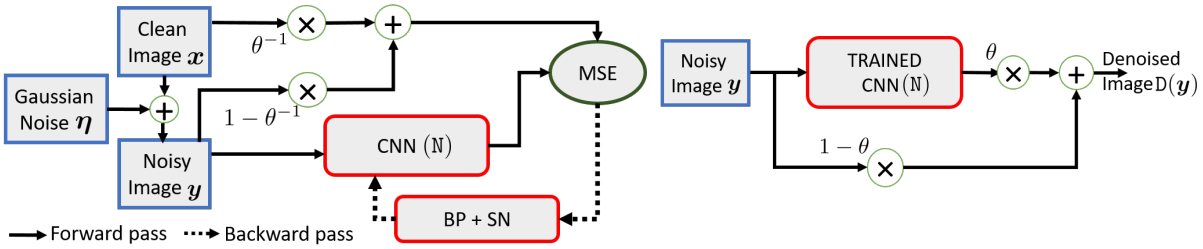
In this section, we focus on the existence of nonlinear denoisers that are averaged on $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$. Indeed, most state-of-the-art denoisers including CNNs are nonlinear. Sufficient conditions for a network to be averaged were formulated in [136]. However, we take a different route—motivated by recent works on the Lipschitz regularization of networks [78, 137, 138, 117], we propose a direct construction of a CNN denoiser which is empirically seen to be θ-averaged for fixed $\theta \in (0, 1)$. The algorithms under consideration are PnP-ISTA (denoiser plugged in ISTA framework in (1.10)), where iterations are given by,

$$\boldsymbol{x}_{k+1} = \mathrm{D}(\boldsymbol{x}_k - \rho^{-1}\nabla f(\boldsymbol{x}_k)), \tag{2.24}$$

and PnP-ADMM (denoiser plugged in ADMM framework in (1.12)), where iterations are given by,

$$\begin{aligned} \boldsymbol{v}_{k+1} &= \mathrm{Prox}_{\rho^{-1}f}(\boldsymbol{x}_k - \boldsymbol{z}_k), \\ \boldsymbol{x}_{k+1} &= \mathrm{D}(\boldsymbol{v}_{k+1} + \boldsymbol{z}_k), \\ \boldsymbol{z}_{k+1} &= \boldsymbol{z}_k + \boldsymbol{v}_{k+1} - \boldsymbol{x}_{k+1}. \end{aligned} \tag{2.25}$$

Note that we have already established convergence results for PnP-ISTA and PnP-ADMM with averaged denoisers in Theorems 2.8 and 2.9.



(a) Training N to be non-expansive. (b) Denoiser $\mathrm{D} = (1-\theta)\mathrm{I} + \theta\mathrm{N}$.

Figure 2.4: The proposed denoiser has the form $\mathrm{D} = (1-\theta)\mathrm{I} + \theta\mathrm{N}$, where N is a CNN. N is trained to minimize the mean-squared-error (MSE) between the prediction $\mathrm{N}(\boldsymbol{y})$ and $(1-\theta^{-1})\boldsymbol{y} + \theta^{-1}\boldsymbol{x}$ over pairs $(\boldsymbol{x}, \boldsymbol{y})$ of clean (target) and noisy (input) images. The backward pass consists of the standard backpropagation (BP)-based gradient update followed by spectral normalization (SN); the latter corrects the weights so that N is empirically seen to be non-expansive.

The schematic of our construction for averaged denoiser is shown in Figure 2.4. We decompose the denoiser into $\mathrm{D} = (1-\theta)\mathrm{I} + \theta\mathrm{N}$, where $\mathrm{N} : \mathbb{R}^n \to \mathbb{R}^n$ has the structure of a CNN and we constrain N to be non-expansive during training. Since $\mathrm{N} = (1-1/\theta)\mathrm{I} + (1/\theta)\mathrm{D}$, if N is trained to be non-expansive, then D is automatically averaged; see Figure 2.4(a). The training mechanism for N is described in Section 2.6.2. After training N, we use it for denoising as shown in Figure 2.4(b).

The network architecture of N, which is described in detail in Section 2.6.1, is identical to that of basic DnCNN [9]. As with DnCNN, N has three types of layers, convolutional, batch normalization, and ReLU activation. The Lipschitz constant of a convolutional layer is the spectral norm (largest singular value $\sigma_{\max}$) of its matrix representation. Moreover, the Lipschitz constant of ReLU is 1 [136]. We constrain each convolutional layer to have $\sigma_{\max} \leqslant 1$. Hence, as in [78, 137], after each gradient update, we ensure that $\sigma_{\max}(\mathbf{M}) \leqslant 1$ by projecting $\mathbf{M}$ onto the space of matrices $\mathbf{P}$ with $\sigma_{\max}(\mathbf{P}) \leqslant 1$, where $\mathbf{M}$ is the matrix representation of a convolutional layer. This can be done using SVD [117] but is expensive since $\mathbf{M}$ is large and this needs to be done for each layer. Hence, we use the fast approximation in [78] called spectral normalization(SN), where the $\sigma_{\max}$ value of a given layer is computed using power iterations and the weights in that layer are scaled by $1/\sigma_{\max}$. Note that unless we remove batch normalization layers, we cannot theoretically guarantee that the trained network is nonexpansive. However, the network is empirically seen to effect the convergence of PnP algorithms as shown in Section 2.7.

After training CNN, which is empirically non-expansive, we add to it an inexpensive projection operation (which is not shown in Figure 2.4(b)). This is essential for the convergence guarantees in Section 2.3; projection operation forces the hypothesis $\mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}}) \neq \varnothing$ in Theorem 2.8 and $\mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}}) \neq \varnothing$ in Theorem 2.9 to be satisfied using Brouwer fixed point theorem [114]. In particular, let C be a closed convex n-cell in $\mathbb{R}^n$ and let $\Pi_C : \mathbb{R}^n \to \mathbb{R}^n$ be

the projection onto C, namely, $\Pi_C(\boldsymbol{x}) = \text{argmin}\,\{||\boldsymbol{z}-\boldsymbol{x}|| : \boldsymbol{z} \in C\}$. We fix the denoisers for PnP-ISTA and PnP-ADMM to be

$$\mathrm{D}^{\star} := \Pi_C \circ \big((1-\theta)\mathrm{I} + \theta \mathrm{N}\big),\ \theta \in (0,1), \quad \text{and} \quad \mathrm{D}_{\star} := \frac{1}{2}(\mathrm{I} + \Pi_C \circ \mathrm{N}). \tag{2.26}$$

Since Π_C is averaged [114], it can be verified that $\mathrm{D}^{\star}$ and $\mathrm{D}_{\star}$ are averaged if N is non-expansive. For PnP-ISTA, we take $C = [0,1]^n$ which is the dynamic range of the (normalized) ground-truth image. For PnP-ADMM, C should be matched with the dynamic range of the output of N, which depends on the imaging application; we empirically found that the choice $C = [-2,2]^n$ gives optimal results for the applications considered in Section 2.5. For all experiments, we have used $\theta = 1/2$ for $\mathrm{D}^{\star}$.

2.6.1 Framework for the CNN Denoiser

The architecture of N is identical to that of DnCNN [9] as shown in Figure 2.5. Thus, N consists of a cascade of blocks where each block has three computational layers: convolution, batch normalization (BN), and ReLU activation. We recall the discription of CNN architecture from Section 1.4.1 for completeness. Mathematically, we can write N as a composition of transforms:

$$\mathrm{N}(\boldsymbol{y};\Theta) = \mathrm{K}_L(\mathrm{R}(\mathrm{B}_{L-2}(\mathrm{K}_{L-1}(\cdots(\mathrm{R}(\mathrm{B}_1(\mathrm{K}_2(\mathrm{R}(\mathrm{K}_1(\boldsymbol{y})))\cdots), \tag{2.27}$$

where

- $\boldsymbol{y} \in \mathbb{R}^{h\times w\times 1}$ is the input image with one channel.
- $L = 17$ is the number of convolutional layers.
- C_ℓ is the number of filters in convolutional layer $\ell \in \{1,\dots L\}$, which is also the number of channels in the output of layer ℓ; $C_\ell = 64$ for $\ell \in \{1,\dots,L-1\}$ and $C_L = 1$.
- For $\ell \in \{1,2,\dots,L\}$, the filters in convolutional layer ℓ are of size $3\times 3\times C_{\ell-1}$.
- Convolutional layer ℓ is represented as a transform $\mathrm{K}_\ell : \mathbb{R}^{h\times w\times C_{\ell-1}} \to \mathbb{R}^{h\times w\times C_\ell}$.
- BN layer $\ell \in \{1,\dots,L-2\}$ is represented as a transform $\mathrm{B}_\ell : \mathbb{R}^{h\times w\times C_{\ell+1}} \to \mathbb{R}^{h\times w\times C_{\ell+1}}$.
- $\mathrm{R}(t) = \max(t,0)$ is the output of the ReLU function which is applied component-wise.
- The hyperparameter Θ are the weights of the convolutional layers and the weights and biases of the BN layers.
- $\mathrm{N}(\boldsymbol{y};\Theta) \in \mathbb{R}^{h\times w\times 1}$ is the output from N with input $\boldsymbol{y}$ and hyperparameter setting Θ.

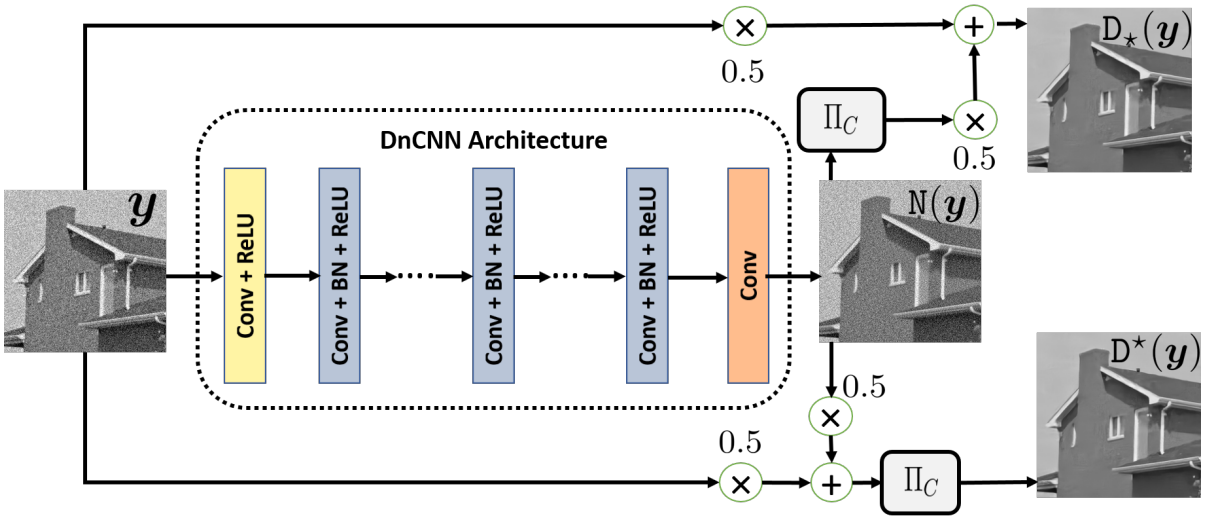


Figure 2.5: Network architecture of N (basic DnCNN). Also shown are the denoisers $\mathrm{D}^\star$ and $\mathrm{D}_\star$ (for PnP-ISTA and PnP-ADMM resp.) built from N and projection operator Π_C.

Let B be the set of weights and bias for all the batch normalization layers. Then the parameter set is,

$$\Theta = \left\{\{\mathrm{K}_\ell\}_{\ell=1}^{L}, \mathrm{B}\right\}$$

2.6.2 Training

We train N on 40×40 patches (which we continue to refer to as images) extracted from images of the BSD500 dataset [139]. Denote the images (normalized to $[0,1^n]$) by $x_1, \ldots, x_m \in \mathbb{R}^n$, where n is the number of pixels in each patch. The noisy images $y_1, \ldots, y_m \in \mathbb{R}^n$ (input to N) are generated by corrupting x_i with additive white Gaussian noise, i.e., $y_i = x_i + \sigma\eta_i$, where $\eta_i \sim \mathcal{N}(0, \mathbf{I})$ and $\sigma \in [0,1]$. The goal is to construct a denoiser $\mathrm{D} : \mathbb{R}^n \to \mathbb{R}^n$ such that $x_i \approx \mathrm{D}(y_i)$ and D is θ-averaged for some fixed $\theta \in (0,1)$. To address these objectives, we decompose the denoiser into $\mathrm{D} = (1-\theta)\mathrm{I} + \theta\mathrm{N}$, where N has the structure of a CNN and is non-expansive. In particular, since $\mathrm{N} = (1-1/\theta)\mathrm{I} + (1/\theta)\mathrm{D}$, the training data for the network is taken to be $\{(y_i, z_i) : 1 \leqslant i \leqslant m\}$, where

$$z_i = (1-1/\theta)y_i + (1/\theta)x_i.$$

The network is trained via the following optimization:

$$\begin{aligned} \min_{\Theta} \quad & \frac{1}{m}\sum_{i=1}^{m} \|\mathrm{N}(y_i, \Theta) - z_i\|_2^2 \\ \text{s.t.} \quad & \|\mathrm{K}_\ell\|_* \leqslant 1 \quad (\ell = 1, \ldots, L). \end{aligned} \tag{2.28}$$

The objective function in (2.28) is nonconvex but the constraints are convex [30, 140].

We can in principle solve (2.28) using projected gradient descent, where the gradient update (using backpropagation [141]) and the projection operation are performed in an alternating fashion. The projection can be computed using SVD [117] but this is expensive. Moreover, the projection has to be performed for all the convolutional layers after the backpropagation step. We need a fast approximation to keep the training time reasonable. We follow [78, 137], where the gradient update is performed using Adam optimizer (using mini-batches of size 128) and the projection is approximated using a method called spectral normalization. At this point, we refer the reader to Fig. 2.4 which depicts the overall training process.

2.6.3 Spectral Normalization

Spectral normalization (SN) was introduced in [137] and later adapted for CNN in [78]. This step is crucial since the filters after the gradient update might violate the property $||\mathrm{K}_\ell||_* \leqslant 1$. For completeness, we explain the main idea behind the SN method in [78]. If $\mathbf{M}_\ell$ is the matrix representation of K_ℓ, then $||\mathrm{K}_\ell||_* = \sigma_{\max}(\mathbf{M}_\ell)$, where $\sigma_{\max}(\mathbf{M}_\ell)$ is the largest singular value of $\mathbf{M}_\ell$. As a result, we can ensure that $||\mathrm{K}_\ell||_* \leqslant 1$ (after the gradient update) via the normalization:

$$\mathrm{K}_\ell \leftarrow \sigma_{\max}(\mathbf{M}_\ell)^{-1}\, \mathrm{K}_\ell.$$

However, the whole point is that we do not store $\mathbf{M}_\ell$ and instead directly work with the convolution filters (masks) associated with K_ℓ; i.e., we treat K_ℓ as a transform which can be implemented efficiently using the associated filters. Thus, the problem reduces to obtaining a good estimate of $\sigma_{\max}(\mathbf{M}_\ell)$ using K_ℓ as a black-box. More specifically, let

$$\boldsymbol{u}_\ell \in \mathbb{R}^{h\times w\times C_\ell} \quad \text{and} \quad \boldsymbol{v}_\ell \in \mathbb{R}^{h\times w\times C_{\ell-1}}$$

be the left and right singular vectors of K_ℓ corresponding to the largest singular value $\sigma_{\max}$, namely, $\mathrm{K}_\ell \boldsymbol{v}_\ell = \sigma_{\max}\boldsymbol{u}_\ell$. Note that we just need the largest singular value and not the complete SVD. This can be done using the classical power method [142]. Namely, starting with some initialization $\boldsymbol{u}_\ell, \boldsymbol{v}_\ell$, we perform the iterations:

$$\boldsymbol{v}_\ell \longleftarrow \frac{\mathrm{K}_\ell^* \boldsymbol{u}_\ell}{||\mathrm{K}_\ell^* \boldsymbol{u}_\ell||_2} \quad \text{and} \quad \boldsymbol{u}_\ell \longleftarrow \frac{\mathrm{K}_\ell \boldsymbol{v}_\ell}{||\mathrm{K}_\ell \boldsymbol{v}_\ell||_2}.$$

where K_ℓ^* is the adjoint of K_ℓ. Importantly, K_ℓ^* can be expressed as a convolution layer using the rotated versions of the filters in K_ℓ and hence can be computed efficiently [143]. At the end of the power iterations, we can estimate $\sigma_{\max}$ using $\langle \boldsymbol{u}_\ell, \mathrm{K}_\ell \boldsymbol{v}_\ell \rangle$, where $\langle \cdot, \cdot \rangle$ is the standard inner-product on $\mathbb{R}^{h\times w\times C_\ell}$. Since the filter weights do not change drastically between consecutive backward passes, the singular vectors in a given pass can be used as a reliable initialization for the power method in the next pass. This is the main idea in SN [78], where

just one iteration of the power method is used in every backward pass to get a good estimate of $\sigma_{\max}$. SN is in fact quite effective in controlling the Lipschitz bound of the network; see Fi. 2.8 in next section and the experiments in [78].

Let $\Theta^\star$ be the optimal hyperparameter obtained by solving (2.28) using the Adam optimizer along with SN. Then $\mathsf{N}(\cdot;\Theta^\star)$ is the required non-expansive map; with some abuse of notation, we refer to $\mathsf{N}(\cdot;\Theta^\star)$ as N. The denoisers $\mathsf{D}_\star$ and $\mathsf{D}^\star$ for PnP-ADMM and PnP-ISTA are constructed from N as shown in Figure 2.5.

2.6.4 Comparison with DnCNN

Since we adapt the original DnCNN denoiser [9], a natural question is does this affect the denoising performance? The results of such a comparison on a couple of standard datasets are shown in Table 2.4. Notice that the constructed denoisers differ from DnCNN by only 0.1-0.2 dB in PSNR and 0.003 in SSIM measures. Thus the performance of the denoisers is comparable to DnCNN.

Table 2.4: Comparison of denoising performance.

Dataset	Method	$\sigma = 15$	$\sigma = 25$	$\sigma = 50$
Set12	DnCNN	32.81/0.9043	30.44/0.8618	27.13/0.7797
	Proposed	32.73/0.9014	30.22/0.8610	27.10/0.7794
BSD68	DnCNN	31.72/0.8948	29.19/0.8319	26.21/0.7188
	Proposed	31.65/0.8947	29.10/0.8316	26.16/0.7186

2.7 Numerical Results for CNN Denoiser

In this section, we report numerical results to support our claim that, along with the empirical convergence guarantee, the proposed PnP algorithms are competitive with state-of-the-art methods for the applications discussed in Section 2.2.4. We also validate the theoretical results and demonstrate the effectiveness of our denoiser (as a regularizer) for image superresolution and despeckling that involve non-strongly convex loss functions. The denoiser is trained on the BSD500 dataset [139] using 40×40 image patches [78]. We follow [9] for initializing the CNN weights and for splitting the dataset into training and validation sets. Similar to [78, 9], we train the denoiser for noise levels in the range $\sigma \in (0.15, 0.2)$. We use the ADAM optimizer with 50 epochs and 128-size mini-batches [144]. The learning rate is 10^{-3} for the first 25 epochs and is then reduced to 10^{-4}. We use the code in [78] for spectral normalization with necessary modifications. The training takes about 5.5 hours on an Nvidia GTX 1080TI GPU. We evaluate the reconstruction quality (with the ground truth as reference) using PSNR and SSIM [145]. We perform superresolution and despeckling experiments on the standard Set12 dataset [9] and report the averaged metrics (Tables 2.5 and 2.6). For comparison with existing methods, we use the original codes shared by authors. We used about 50 PnP iterations (outer loops) for all experiments.

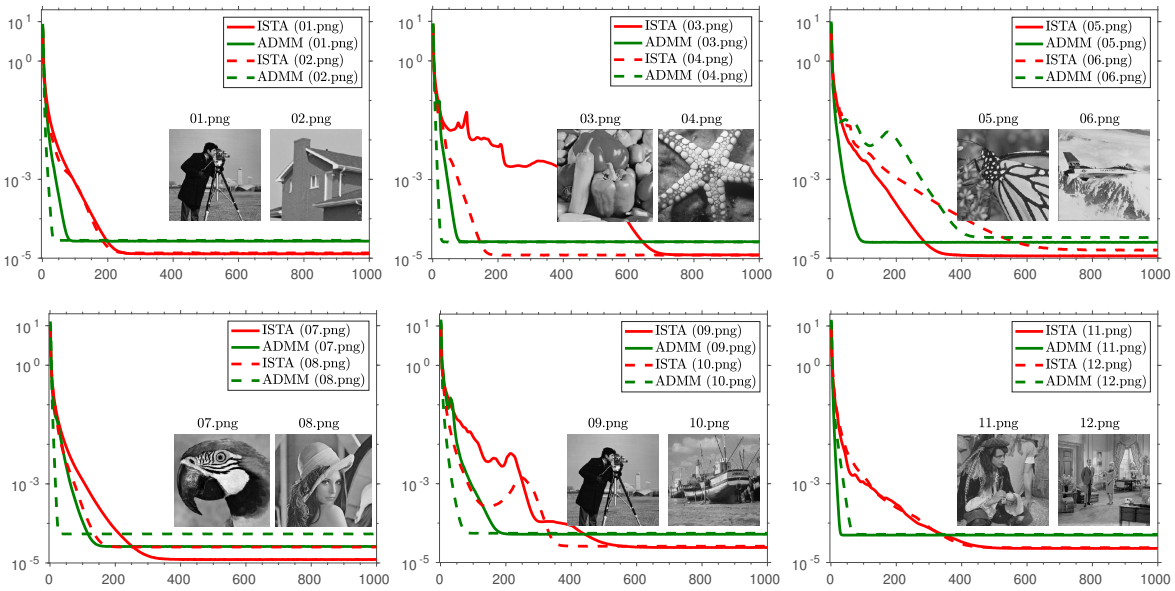


Figure 2.6: Decay of the residual $||x_{k+1} - x_k||_2$ for PnP-ISTA and PnP-ADMM for deblurring a Gaussian blur of size 9×9 and standard deviation 1 (noise level $\sigma = 0.04$).

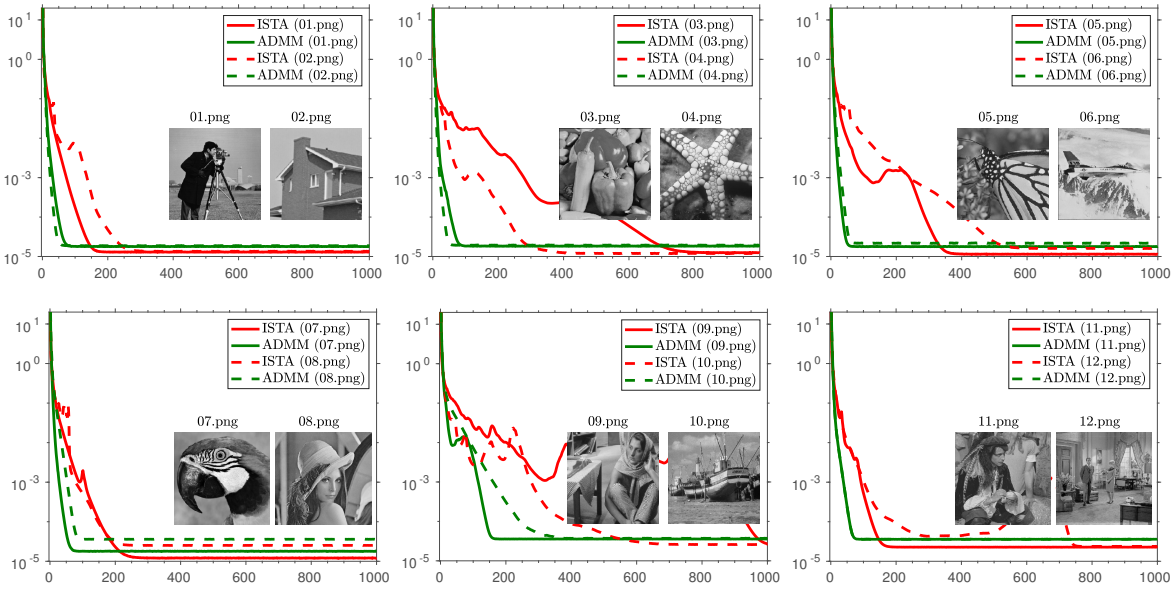


Figure 2.7: Same as in Figure 2.6 but for 2×-superresolution. The experimental setup is identical to that in Table 2.5.

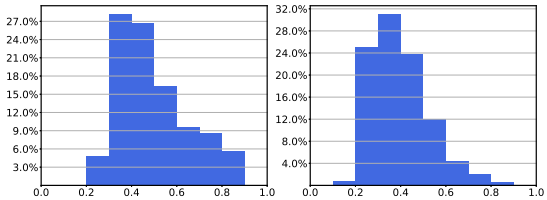


Figure 2.8: Histogram of the ratio $||\mathrm{N}(x_1) - \mathrm{N}(x_2)||_2/||x_1 - x_2||_2$ over images x_1, x_2 from the BSD68 dataset [139] (with noise added); N is trained at $\sigma = 15$ (left) and 30 (right).

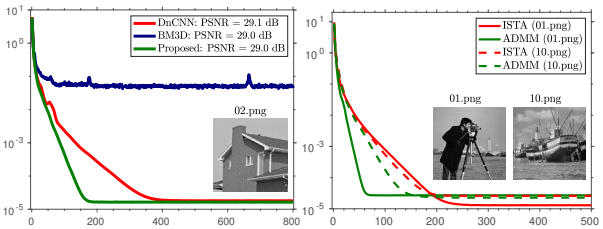


Figure 2.9: Plot of the residual $||x_{k+1} - x_k||_2$ with iterations k for PnP-ISTA and PnP-ADMM (with different denoisers) for 2× image superresolution.

Validation. We first verify that the trained network N in Section 2.6 is non-expansive. In particular, we check that the ratio $||\mathrm{N}(x_1) - \mathrm{N}(x_2)||_2/||x_1 - x_2||_2$ is in the range $[0, 1]$ for different (noisy) image pairs (x_1, x_2) sampled from BSD68 dataset [139]. This is indeed the case as shown in Figure 2.8. To ascertain empirical convergence of the iterates generated by PnP-ISTA and PnP-ADMM (Theorems 2.8 and 2.9), we plot the residuals (see Figure 2.9). In the first plot, we have used PnP-ADMM with different denoisers. Notice that the residual does not seem to decay with the BM3D denoiser; this is necessary for the iterates to converge though not sufficient. On the other hand, we notice that the decay is quite fast with our averaged denoiser (also for the modified DnCNN in [78]). In the second plot, we notice that convergence is faster for ADMM compared to ISTA with our denoiser (this is often true with classical regularizers). We next show more convergence results to show that analysis is application and settings independent. In particular, we plot the residual $||x_{k+1} - x_k||_2$ for PnP-ISTA and PnP-ADMM in Figures 2.6 and 2.7 for deblurring and superresolution. Note that convergence of the residual to zero is necessary for the iterates to converge though not sufficient. As in Figure 2.9, we notice that convergence is faster for ADMM compared to ISTA.

Table 2.5: Comparison for superresolution.

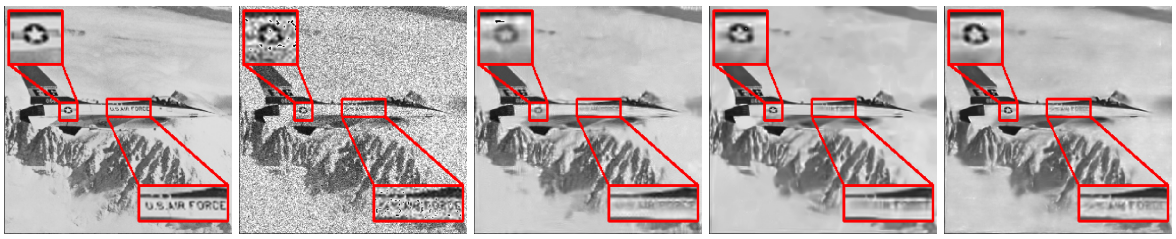
						PnP-ADMM	
	SR[146]	GPR[147]	SPSR[148]	TSE[149]	NCSR[123]	BM3D	Proposed
$K = 2$	25.98	26.68	26.29	25.93	28.10	29.25	**29.36**
$K = 4$	20.95	22.20	21.21	20.92	24.64	**25.98**	25.89

Superresolution. We compare the superresolution results of PnP-ADMM (using our denoiser) with existing methods that can handle the degradation model described in Section 2.2.4. The results are reported in Table 2.5 for the dataset in [100]; the PSNR values are averaged over the images in the dataset. For all the experiments, we use a symmetric Gaussian blur of size 9×9 and standard deviation 1, and η is Gaussian noise with variance $5/255$. Apart from PnP-ADMM and NCSR, the reported results are from [100]. See Figure 2.10 for a visual comparison.

(a) Ground truth (b) TV [150] (c) BM3D [70] (d) DnCNN [78] (e) Proposed

Figure 2.10: 2×-image superresolution using PnP-ISTA (20 iterations) with different denoisers. Compared to BM3D, the visual quality is perceivably better in relation to the ground truth for CNN denoisers (compare the zoomed sections). The quality metrics are: (b) PSNR: 24.3 dB, SSIM: 0.83; (c) PSNR: 26.2 dB, SSIM: 0.86; (d) PSNR: 27.4 dB, SSIM: 0.89; (e) PSNR: 27.3 dB, SSIM: 0.90.

Despeckling. For despeckling, since ∇f is not Lipschitz continuous, we do not use PnP-ISTA (see Theorem 2.8). Instead, we use PnP-ADMM using the proposed CNN denoiser (and other denoisers) as regularizer and compare the results with state-of-the-art methods [5, 122, 151]; see Table 2.6. A visual reconstruction is shown in Figure 2.11 for $M = 5$ looks (compare the zoomed sections).



(a) Ground-truth (b) Observed (c) [151] (d) [122] (e) Proposed

Figure 2.11: Despeckling results. The reconstruction for our method is visually better. The PSNR, SSIM values are: (b) 18 dB, 0.41; (c) 26.2 dB, 0.811; (d) 26.4 dB, 0.813; (e) 26.5 dB, 0.80.

Table 2.6: Despeckling using PnP-ADMM and comparison with existing algorithms.

						PnP-ADMM		
		Noisy	[5]	[151]	[122]	BM3D	[78]	Proposed
$M = 3$	PSNR	16.87	25.18	27.71	27.96	27.56	27.40	27.32
	SSIM	0.34	0.70	0.80	0.81	0.79	0.77	0.78
$M = 5$	PSNR	18.90	26.57	28.89	28.92	28.71	28.87	28.72
	SSIM	0.41	0.74	0.84	0.84	0.82	0.82	0.83
$M = 10$	PSNR	21.70	27.48	30.50	30.88	30.35	30.55	30.50
	SSIM	0.51	0.78	0.87	0.87	0.86	0.86	0.86

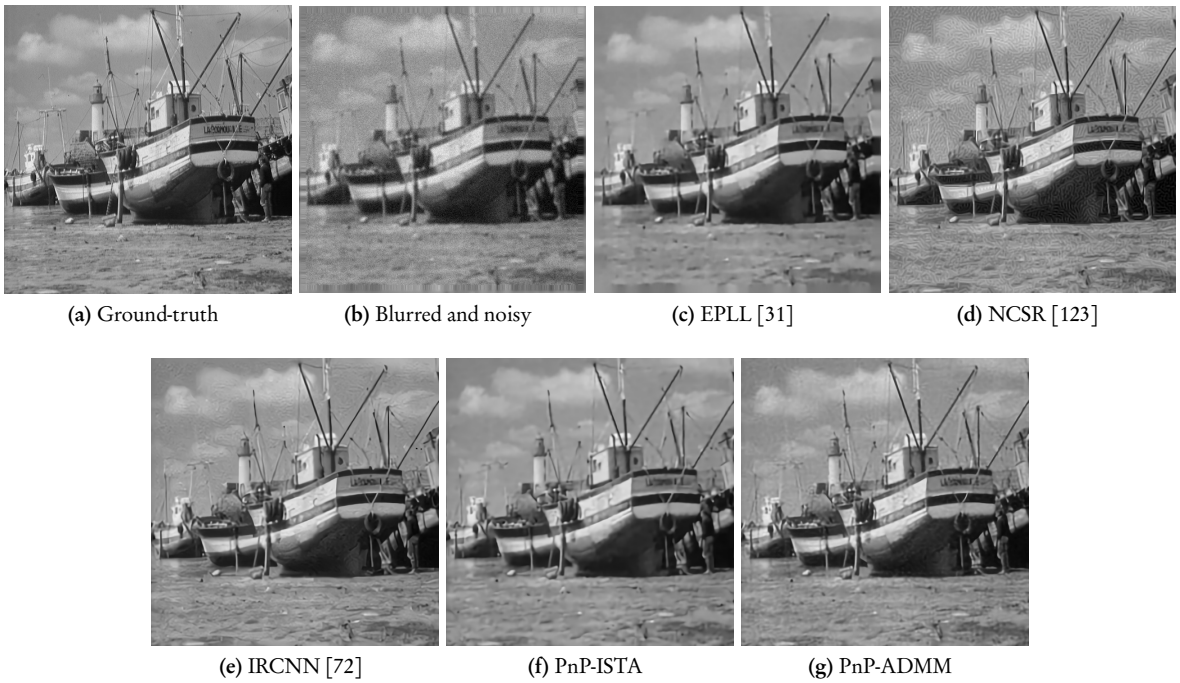
Deblurring. To highlight the state-of-the-art performance for deblurring, we perform three standard experiments introduced in [152] with the following settings of Gaussian blur and noise level:

Exp-I: Blur of size 25×25 and standard deviation 1.6, and $\sigma = 0.04$.

Exp-II: Blur of size 25×25 and standard deviation 1.6, and $\sigma = 2/255$.

Exp-III: Blur of size 25×25 and standard deviation 3, and $\sigma = 0.04$.

An extensive comparison is shown in Table 2.7. The proposed PnP algorithms and IRCNN [72] are found to be top-performing. Apart for the last four methods in Table 2.7, the results are straight from [101]. PSNR values are averaged over the dataset in [152]. A visual comparison is shown in Figure 2.12. We notice that the reconstructed image for our method (and IRCNN) is not over-smoothed and does not have artefacts; this also corroborates in the PSNR values in Table 2.7.



(a) Ground-truth **(b)** Blurred and noisy **(c)** EPLL [31] **(d)** NCSR [123]

(e) IRCNN [72] **(f)** PnP-ISTA **(g)** PnP-ADMM

Figure 2.12: Deblurring (Exp-I) using PnP-ISTA and PnP-ADMM with the proposed denoisers and comparison with state-of-the-art methods. PSNR, SSIM values: (b) 24.5 dB, 0.4452; (c) 26.31 dB, 0.7045; (d) 26.90 dB, 0.695; (e) 27.93 dB, 0.746; (f) 27.59 dB, 0.753; (g) 27.98 dB, 0.763.

Inpainting. We perform the three standard experiments introduced in [87]:

Exp-I: 80 percent missing pixels and $\sigma = 0$ (noiseless).

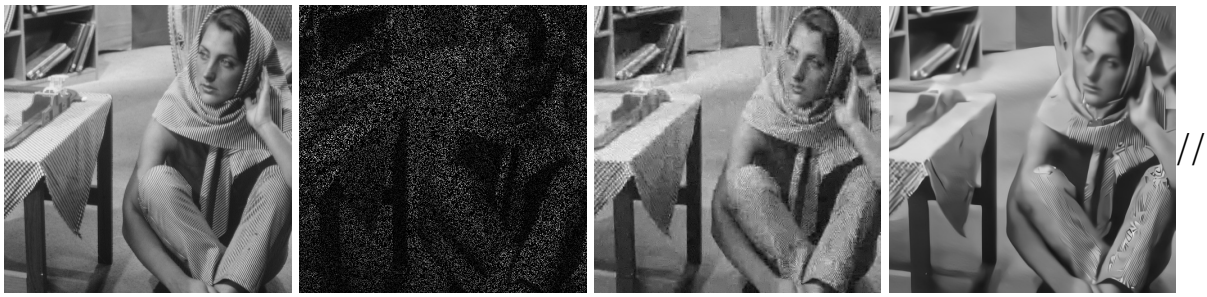
Exp-II: 80 percent missing pixels and $\sigma = 10/255$.

Exp-III: 80 percent missing pixels and $\sigma = 12/255$.

The results of these experiments are shown in Tables 2.8, 2.9 and 2.10. Except for EPLL, the results reported in the tables are from [87]. A visual comparison is shown in Figure 2.13. It is evident from these results that we are competitive with recent state-of-the-art methods for inpainting.

Table 2.7: Comparison for deblurring.

Method	Exp-I	Exp-II	Exp-III
EPLL [31]	24.04	26.64	21.36
IRLS [153]	24.09	26.51	21.72
LUT [154]]	24.17	26.60	21.73
DEB-BM3D [70]	24.19	26.30	21.48
IDD-BM3D [155]	24.68	27.13	21.99
FoE [156]	24.07	26.56	21.61
MLP [152]	24.76	27.23	22.20
FlexISP [128]	24.32	26.84	21.99
[101]	24.51	27.08	21.83
NCSR [123]	26.62	30.03	24.51
IRCNN [72]	**27.82**	**30.62**	**25.16**
PnP-ISTA	27.26	27.50	23.85
PnP-ADMM	**27.72**	**30.30**	**25.01**



(a) Ground truth **(b)** Observed **(c)** EPLL [31] **(d)** IDBP-CNN [87]

(e) IDBP-BM3D [87] **(f)** PnP-ISTA **(g)** PnP-ADMM

Figure 2.13: Inpainting (Exp-II) using PnP-ADMM and comparison with state-of-the-art methods. PSNR, SSIM values: (c) 24.45 dB, 0.688; (d) 24.50 dB, 0.727; (e) 24.62 dB, 0.713; (f) 24.10 dB, 0.667; (g) 26.58 dB, 0.812.

Table 2.8: Experiment 1: Performance comparison.

Method	cameraman	house	peppers	lena	barbara	boat	hill	couple
EPLL [31]	24.31 / 0.816	30.71 / 0.867	25.92 / 0.854	30, 42 / 0.867	25.43 / 0.792	27.51 / 0.788	29.06 / 0.776	27.69 / 0.801
IPPO [124]	24.78 / 0.832	32.64 / 0.909	28.22 / 0.882	31.84 / 0.895	**29.89** / **0.906**	28.17 / 0.822	29.47 / 0.815	28.22 / 0.842
[79]	24.83 / 0.845	**34.72** / **0.920**	28.77 / 0.895	32.41 / **0.903**	25.68 / 0.862	28.83 / 0.844	**29.95** / 0.831	29.01 / **0.865**
IRCNN[72]	25.27 / 0.838	32.21 / 0.888	28.26 / 0.882	31.56 / 0.889	27.34 / 0.858	27.88 / 0.809	29.24 / 0.804	28.23 / 0.834
IDBP-BM3D [87]	24.86 / 0.840	33.78 / 0.893	28.58 / 0.885	32.13 / 0.893	25.55 / 0.841	28.51 / 0.824	29.74 / 0.810	28.80 / 0.846
IDBP-CNN [87]	24.24 / 0.826	32.14 / 0.881	27.80 / 0.866	31.22 / 0.880	24.29 / 0.796	27.72 / 0.803	29.01 / 0.790	27.98 / 0.818
PnP-ISTA	24.54 / 0.784	30.79 / 0.834	27.39 / 0.812	30.37 / 0.829	24.28 / 0.722	27.15 / 0.727	28.01 / 0.701	27.25 / 0.738
PnP-ADMM	**25.94** / **0.864**	33.22 / 0.893	**29.45** / **0.897**	**32.66** / 0.900	28.67 / 0.894	**29.11** / **0.846**	29.61 / **0.834**	**29.32** / 0.861

Table 2.9: Experiment 2: Performance comparison.

Method	cameraman	house	peppers	lena	barbara	boat	hill	couple
EPLL [31]	23.88 / 0.727	29.10 / 0.773	25.06 / 0.776	28.95 / 0.777	24.76 / 0.720	26.59 / 0.720	27.86 / 0.710	26.70 / 0.733
[79]	24.55 / 0.785	**31.53** / **0.848**	27.21 / 0.827	30.10 / 0.833	24.45 / 0.735	27.01 / 0.731	27.94 / 0.706	27.23 / 0.761
IRCNN [72]	24.75 / 0.794	30.61 / 0.845	27.25 / 0.827	29.94 / 0.833	25.94 / 0.781	26.86 / 0.741	27.90 / 0.726	26.98 / 0.757
IDBP-BM3D [87]	24.68 / 0.786	31.62 / 0.850	27.24 / 0.829	30.14 / 0.835	25.03 / 0.755	27.02 / 0.731	28.00 / 0.708	27.22 / 0.759
IDBP-CNN [87]	23.94 / 0.791	31.16 / 0.851	27.25 / 0.841	30.17 / **0.849**	23.62 / 0.753	26.95 / 0.756	27.93 / 0.734	27.04 / 0.773
PnP-ISTA	24.61 / 0.791	30.94 / 0.836	27.18 / 0.815	29.94 / 0.829	23.95 / 0.665	26.95 / 0.730	27.81 / 0.705	26.96 / 0.789
PnP-ADMM	**25.31** / **0.807**	31.36 / 0.841	**28.06** / **0.853**	**30.38** / 0.840	**26.65** / **0.816**	**27.64** / 0.760	**28.47** / **0.749**	**27.72** / **0.785**

Table 2.10: Experiment 3: Performance comparison.

Method	cameraman	house	peppers	lena	barbara	boat	hill	couple
EPLL [31]	23.64 / 0.691	28.44 / 0.732	24.75 / 0.741	28.32 / 0.737	24.45 / 0.689	26.18 / 0.689	27.33 / 0.679	26.26 / 0.724
[79]	24.43 / 0.774	30.78 / 0.839	26.56 / 0.807	29.47 / 0.818	24.12 / 0.705	26.53 / 0.707	27.44 / 0.683	26.71 / 0.734
IRCNN [72]	**24.59** / 0.781	30.19 / 0.835	26.94 / 0.813	29.52 / 0.820	25.49 / 0.758	26.58 / 0.723	27.55 / 0.706	26.62 / 0.736
IDBP-BM3D [87]	24.51 / 0.775	**31.14** / **0.844**	26.79 / 0.816	29.69 / 0.824	25.06 / 0.738	26.64 / 0.712	27.61 / 0.691	26.77 / 0.738
IDBP-CNN [87]	24.14 / 0.786	30.92 / 0.843	27.17 / **0.830**	29.80 / **0.836**	23.61 / 0.731	26.78 / 0.738	27.70 / 0.714	26.80 / 0.752
PnP-ISTA	24.56 / **0.790**	30.38 / 0.831	26.78 / 0.814	29.90 / 0.830	23.88 / 0.716	26.86 / 0.731	27.65 / 0.703	27.04 / 0.744
PnP-ADMM	24.41 / 0.786	31.10 / 0.822	**27.48** / **0.830**	**30.16** / 0.829	**26.39** / **0.793**	**27.20** / **0.744**	**27.98** / **0.727**	**27.09** / **0.762**

The (average) parameter settings for different experiments are reported in Table 2.11. To get the best results for a given input image, we need to fine-tune the parameters around these values, but the parameters in Table 2.11 are close to the optimal parameters.

Table 2.11: Parameters used for different experiments.

Application		PnP-ISTA		PnP-ADMM	
		σ	ρ	σ	ρ
Superresoltion	$K=2$	5	4.0	15	0.05
	$K=4$	20	20.0	25	0.05
Despeckling	$M=3$	-	-	15	1.5
	$M=5$	-	-	15	5
	$M=10$	-	-	15	5
Deblurring	Exp-I	5	1.0	35	0.05
	Exp-II	5	1.0	30	0.005
	Exp-III	5	1.0	35	0.05
Inpainting	Exp-I	15	2.0	20	0.01
	Exp-II	15	2.0	30	0.01
	Exp-III	20	2.0	30	0.01

2.8 Comparison of Denoisers in 2.4 and 2.6

In this section, we compare PnP-ADMM with kernel denoiser (nonlocal means) proposed in Section 2.4 and PnP-ADMM with CNN denoiser (empirically seen to be averaged) proposed in Section 2.6. We refer Table 2.12 for brief comparison of the same.

In Fig. 2.14 and Table 2.13, we present a comparison of PnP regularization capabilities using NLM denoiser and CNN denoiser constructed in Section 2.6. The application is superresolution where measurements are highly degraded; downsampling factor of 4 and additive Gaussian noise added. Note that this is just a sample result where CNN denoiser in 2.6 is better than the linear denoisers. The reconstruction examples where NLM denoiser

Property	Kernel denoiser	Averaged CNN denoiser	Comments
Theoretical convergence of PnP algorithms	Yes	No	CNN denoiser constructed in not provably averaged due to batch normalization layers and usage of power iterations for spectral normalization (refer Chapter 4).
Less Inference time	No	Yes	Time taken for the application of NLM denoiser is 3× that of CNN denoiser.
Superior Regularization capabilities	No	Yes	In many reconstruction examples, CNN denoiser outperforms NLM denoiser within PnP algorithms.

Table 2.12: Detailed comparison of kernel denoiser NLM and constructed Averaged CNN denoiser for PnP regularization.

outperforms CNN denoiser for regularization are fewer in number and shown in Chapter 3.

(a) Ground Truth. **(b)** y **(c)** Upsampled input. **(d)** NLM(PSNR 23.13dB). **(e)** CNN(PSNR 24.88dB).

Figure 2.14: Superresolution with downsampling factor 4 and additive Gaussian noise. When degradation is higher, proposed CNN denoiser is shown to obtain better regularization capabilities than kernel denoiser NLM.

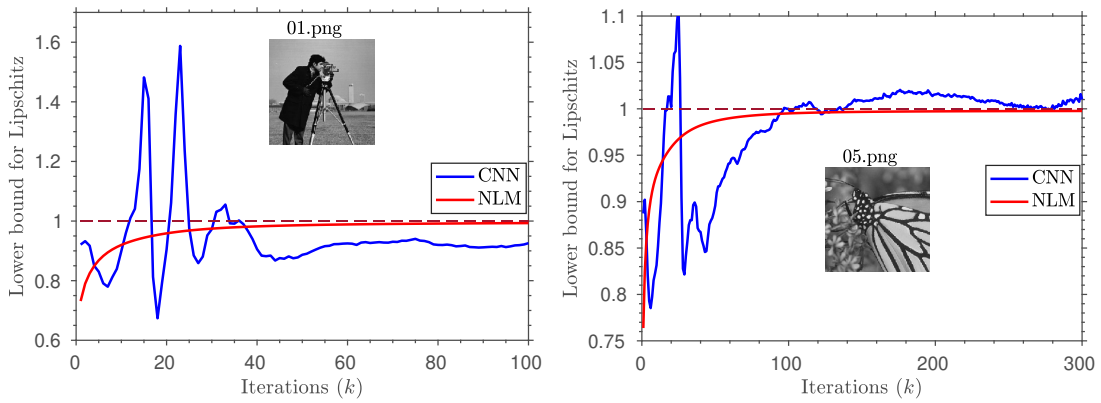
	NLM	CNN
PSNR	24.54	25.30
SSIM	0.7444	0.7529

Table 2.13: PSNR and SSIM comparison for superresolution experiment in Fig. 2.14 averaged over Set 12 dataset.

Though the CNN denoiser in Section 2.6 demonstrates better regularization capabilities than kernel denoisers, there are pathological examples where the constructed CNN denoiser does not satisfy the necessary condition of nonexpansivity as shown in Fig. 2.15. Note that if we remove batch normalization layers, the denoising accuracy is off by around 3 dB and still the constructed denoiser need not be nonexpansive as shown in Chapter 4, possibly due to the usage of power iterations for spectral normalization.

2.9 Conclusion

We showed that averaged operators are sufficient for convergence of PnP-ISTA and PNP-ADMM updates. Importantly, iterate and objective convergence of PnP-ISTA and PnP-ADMM is guaranteed for a class of linear denoisers provided we work with a denoiser-specific



(a) CNN denoiser in Section 2.6 **(b)** NLM denoiser

Figure 2.15: Violation of nonexpansive property: For superresolution for downsampling factor of 4, PnP-ADMM is run for 500 iterations by plugging different denoisers and lower bound for Lipschitz is plotted against iterations. Lipschitz is defined as supremum of $\|\mathrm{D}(\boldsymbol{u})-\mathrm{D}(\boldsymbol{v})\|_2/\|\boldsymbol{u}-\boldsymbol{v}\|_2$, for all possible images $\boldsymbol{u}$ and $\boldsymbol{v}$. Thus by considering successive iterates $\boldsymbol{x}_k$ as $\boldsymbol{u}$ and $\boldsymbol{x}_{k+1}$ as $\boldsymbol{v}$, various estimates of lower bound for Lipschitz are obtained. Note that for CNN denoiser in Section 2.6, lower bound for Lipschitz is greater than 1 for many iterations, thereby proving that actual Lipschitz is greater than 1 and hence denoiser is not provably averaged. Whereas, for the same set of inputs, lower bound for Lipschitz in case of NLM is always strictly less than 1 Note the subtle difference that the inner product considered for CNN is standard inner product $\langle\cdot,\cdot\rangle_2$ but the inner product considered for NLM is denoiser specific $\langle\cdot,\cdot\rangle$.

inner product (and associated gradient and proximal operators). Importantly, our analysis holds for non-symmetric kernel filters like nonlocal means which is known to possess good regularization capabilities. In fact, we demonstrated this for model-based superresolution and despeckling. We also proposed a direct construction of a CNN denoiser that is seen to be empirically averaged and obtain state-of-the-art reconstruction performance for various applications within PnP-ISTA and PnP-ADMM. We then compare the regularization capabilities of linear denoiser and our constructed averaged denoiser and discuss the respective open problems, which indeed is the motivation for Chapters 3 and 4. The following are the limitaions of the algorithms proposed in this chapter.

PnP algorithms in (2.11) and 2.12 with kernel denoiser: One application of kernel denoiser is computationally more complex than applying a CNN-based denoiser. This is because for obtaining the denoised output at any pixel $\boldsymbol{s}$ we need to calculate weights $\phi(\zeta_s,\zeta_t)$ for all neighbour pixels $\boldsymbol{t}\in\Omega$. Hence, we need algorithms with better convergence rate (lesser number of iterations) than PnP algorithms, when we use kernel denoisers for regularization. Such an algorithm is proposed in Chapter 3.

PnP algorithms with averaged CNN denoiser: It is clear from Fig. 2.14 and Table 2.13 that learning-based denoiser constructed in Section 2.6 demonstrates better regularization capabilities than kernel denoiser in many reconstruction examples (vice versa is also true in some settings as shown in Chapter 3). At the same time, there are pathological examples as in Fig. 2.15, which infer that the CNN denoiser constructed in Section 2.6 is not provably

nonexpansive. Hence, we could possibly find examples as shown in Table 1.3 where PnP algorithms may diverge resulting in spurious reconstructions though we have not seen any such cases for our constructed denoiser. Hence, we require learning-based denoisers which are provably averaged and have excellent regularization capabilities. Such a learning-based denoiser is proposed in Chapter 4.

2.10 Appendix

In this section, we give detailed proofs of the results in Section 2.3. Unless specified otherwise, it should be understood that we work in $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$, where $\langle\cdot,\cdot\rangle$ is an arbitrary inner product and $||\cdot||$ denotes the norm induced by $\langle\cdot,\cdot\rangle$.

2.10.1 Proof of Proposition 2.4

(a) Let $\mathbf{A} = (1/\theta)\mathbf{W} - (1/\theta - 1)\mathbf{I}$ and $\theta \in [1/2, 1)$. Since the eigenvalues of $\mathbf{W}$ are in $[0,1]$, the eigenvalues of $\mathbf{A}$ must be in $[-1,1]$. Since $\mathbf{A}$ is symmetric, this means that its spectral norm $||\mathbf{A}||_2$ (largest singular value) is at most 1. Hence, $||\mathbf{A}(\boldsymbol{x}-\boldsymbol{y})||_2 \leqslant ||\mathbf{A}||_2 ||\boldsymbol{x}-\boldsymbol{y}||_2 \leqslant ||\boldsymbol{x}-\boldsymbol{y}||_2$, i.e., $\mathbf{A}$ is non-expansive. Hence, $\mathbf{W}$ is θ-averaged on $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_2)$.

(b) Since $\mathbf{W}$ is row-stochastic, $\mathbf{W}\boldsymbol{e} = \boldsymbol{e}$, where $\boldsymbol{e} \in \mathbb{R}^n$ is the all-ones vector. Thus, 1 is an eigenvalue of $\mathbf{W}$. Since $||\mathbf{W}||_2 = \max\{||\mathbf{W}\boldsymbol{x}||_2 : ||\boldsymbol{x}||_2 = 1\}$, we can conclude that $||\mathbf{W}||_2 \geqslant 1$.

Suppose that D is averaged. Then we can show that $\mathbf{W}^\top \boldsymbol{e} = \boldsymbol{e}$. But this would contradict our assumption that $\mathbf{W}$ is not doubly stochastic, hence D cannot be averaged. Indeed, if D is averaged, then it is non-expansive: $||\mathbf{W}\boldsymbol{x}||_2 \leqslant ||\boldsymbol{x}||_2$ for all $\boldsymbol{x} \in \mathbb{R}^n$. Thus, we must have $||\mathbf{W}||_2 \leqslant 1$. Combined with the fact $||\mathbf{W}||_2 \geqslant 1$, we get $||\mathbf{W}||_2 = 1$. Since $\left\|\mathbf{W}^\top\right\|_2 = ||\mathbf{W}||_2$,

$$\begin{aligned} ||\boldsymbol{e}||_2^2 = \boldsymbol{e}^\top(\mathbf{W}\boldsymbol{e}) = (\mathbf{W}^\top\boldsymbol{e})^\top\boldsymbol{e} &\leqslant \left\|\mathbf{W}^\top\boldsymbol{e}\right\|_2 ||\boldsymbol{e}||_2 \\ &\leqslant \left\|\mathbf{W}^\top\right\|_2 ||\boldsymbol{e}||_2^2 = ||\boldsymbol{e}||_2^2 . \end{aligned}$$

Thus, both "$\leqslant$" are in fact "$=$". In particular, for the Cauchy-Schwarz inequality, we must have $\mathbf{W}^\top\boldsymbol{e} = \alpha\boldsymbol{e}$, where $\alpha \in \mathbb{R}$. However, $(\alpha\boldsymbol{e})^\top\boldsymbol{e} = (\mathbf{W}^\top\boldsymbol{e})^\top\boldsymbol{e} = \boldsymbol{e}^\top(\mathbf{W}\boldsymbol{e}) = \boldsymbol{e}^\top\boldsymbol{e}$. Hence, $\alpha = 1$ and $\mathbf{W}^\top\boldsymbol{e} = \boldsymbol{e}$, as claimed.

2.10.2 Proof of Lemma 2.6

To prove Lemma 2.6, we need the following result. A proof can be found in [114, Proposition 4.35(iii)]. Nevertheless, we offer a more self-contained analysis.

Lemma 2.17. *Let* $\mathrm{T} : \mathbb{R}^n \to \mathbb{R}^n$ *be* θ*-averaged for some* $\theta \in (0,1)$*. Then, for all* $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$*,*

$$\begin{aligned} ||\mathrm{T}(\boldsymbol{x}) - \mathrm{T}(\boldsymbol{y})||^2 \leqslant &||\boldsymbol{x}-\boldsymbol{y}||^2 \\ &- \frac{1-\theta}{\theta} ||(\mathrm{I}-\mathrm{T})(\boldsymbol{x}) - (\mathrm{I}-\mathrm{T})(\boldsymbol{y})||^2 . \end{aligned}$$

Proof. Let $\mathrm{T} = (1-\theta)\mathrm{I} + \theta\mathrm{S}$, where S is non-expansive. Then $\mathrm{I}-\mathrm{T} = \theta(\mathrm{I}-\mathrm{S}) = \theta\Delta\mathrm{S}$, where $\Delta\mathrm{S} = \mathrm{I}-\mathrm{S}$. We need to show that for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$,

$$||\mathrm{T}(\boldsymbol{x})-\mathrm{T}(\boldsymbol{y})||^2 \leqslant ||\boldsymbol{x}-\boldsymbol{y}||^2 - \theta(1-\theta)\,||\Delta\mathrm{S}(\boldsymbol{x})-\Delta\mathrm{S}(\boldsymbol{y})||^2 .$$

Now $\mathrm{T}(\boldsymbol{x})-\mathrm{T}(\boldsymbol{y}) = (\boldsymbol{x}-\boldsymbol{y}) - \theta\big(\Delta\mathrm{S}(\boldsymbol{x})-\Delta\mathrm{S}(\boldsymbol{y})\big)$. Hence,

$$\begin{aligned} &||\mathrm{T}(\boldsymbol{x})-\mathrm{T}(\boldsymbol{y})||^2 = \\ &\big[||\boldsymbol{x}-\boldsymbol{y}||^2 - \theta(1-\theta)\,||\Delta\mathrm{S}(\boldsymbol{x})-\Delta\mathrm{S}(\boldsymbol{y})||^2\big] \\ &+\big[\theta\,||\Delta\mathrm{S}(\boldsymbol{x})-\Delta\mathrm{S}(\boldsymbol{y})||^2 - 2\theta\,\langle \boldsymbol{x}-\boldsymbol{y}, \Delta\mathrm{S}(\boldsymbol{x})-\Delta\mathrm{S}(\boldsymbol{y})\rangle\big]. \end{aligned}$$

It suffices to show that the second term on the right is negative. Indeed, note that

$$\begin{aligned} &||\Delta\mathrm{S}(\boldsymbol{x})-\Delta\mathrm{S}(\boldsymbol{y})||^2 - 2\,\langle \boldsymbol{x}-\boldsymbol{y}, \Delta\mathrm{S}(\boldsymbol{x})-\Delta\mathrm{S}(\boldsymbol{y})\rangle \\ &= \langle \Delta\mathrm{S}(\boldsymbol{x})-\Delta\mathrm{S}(\boldsymbol{y}), \Delta\mathrm{S}(\boldsymbol{x})-\Delta\mathrm{S}(\boldsymbol{y}) - 2(\boldsymbol{x}-\boldsymbol{y})\rangle \\ &= \langle \boldsymbol{x}-\boldsymbol{y}-\mathrm{S}(\boldsymbol{x})+\mathrm{S}(\boldsymbol{y}), -\boldsymbol{x}+\boldsymbol{y}-\mathrm{S}(\boldsymbol{x})+\mathrm{S}(\boldsymbol{y})\rangle \\ &= ||\mathrm{S}(\boldsymbol{y})-\mathrm{S}(\boldsymbol{x})||^2 - ||\boldsymbol{y}-\boldsymbol{x}||^2 \leqslant 0, \end{aligned}$$

since S is non-expansive. □

We now establish Lemma 2.6 using the above result. Letting $\mathrm{S} = \mathrm{I}-\mathrm{T}$, note that $\boldsymbol{x}_* \in \mathrm{fix}(\mathrm{T})$ if and only if $\mathrm{S}(\boldsymbol{x}_*) = 0$. Since by assumption $\mathrm{fix}(\mathrm{T}) \neq \varnothing$, let $\boldsymbol{x}_* \in \mathrm{fix}(\mathrm{T})$. Setting $\boldsymbol{x} = \boldsymbol{x}_k$ and $\boldsymbol{y} = \boldsymbol{x}_*$ in Lemma 2.17, we have

$$\left\|\boldsymbol{x}_{k+1}-\boldsymbol{x}_*\right\|^2 \leqslant ||\boldsymbol{x}_k-\boldsymbol{x}_*||^2 - \frac{1-\theta}{\theta}\,||\mathrm{S}(\boldsymbol{x}_k)||^2, \tag{2.29}$$

since $\mathrm{S}(\boldsymbol{x}_*) = 0$. By telescoping the sum in (2.29), we obtain

$$\left\|\boldsymbol{x}_{k+1}-\boldsymbol{x}_*\right\|^2 \leqslant ||\boldsymbol{x}_0-\boldsymbol{x}_*||^2 - \frac{1-\theta}{\theta}\sum_{j=0}^{k}\left\|\mathrm{S}(\boldsymbol{x}_j)\right\|^2 .$$

The quantity on the right is nonnegative for all $k \geqslant 1$. In particular, the series $\sum_{j=0}^{\infty}\left\|\mathrm{S}(\boldsymbol{x}_j)\right\|^2$ is bounded above by $||\boldsymbol{x}_0-\boldsymbol{x}_*||^2$. Thus, we must have $\mathrm{S}(\boldsymbol{x}_k) \to 0$ as $k \to \infty$.

On the other hand, it follows from (2.29) that

$$\left\|\boldsymbol{x}_{k+1}-\boldsymbol{x}_*\right\| \leqslant ||\boldsymbol{x}_k-\boldsymbol{x}_*|| \leqslant \cdots \leqslant ||\boldsymbol{x}_0-\boldsymbol{x}_*|| . \tag{2.30}$$

We conclude that $(\boldsymbol{x}_k)_{k\geqslant 0}$ is bounded and thus must have a convergent subsequence, i.e., a

subsequence $(x_{k_r})_{r\geqslant 0}$ that converges to some $\hat{x}\in\mathbb{R}^n$. Since S is continuous, we have

$$S(\hat{x}) = \lim_{r\to\infty} S(x_{k_r}) = \lim_{k\to\infty} S(x_k) = 0,$$

Hence, $\hat{x}\in \mathrm{fix}(T)$.

We are done if we can show that the original sequence $(x_k)_{k\geqslant 0}$ converges to $\hat{x}$. Now, given any $\epsilon > 0$, we can find $N \geqslant 1$ such that $\left\|x_{k_n} - \hat{x}\right\| < \epsilon$ for $n \geqslant N$. Let $K = k_N$. Since $\hat{x}\in\mathrm{fix}(T)$, it follows from (2.30) that for $k \geqslant K$,

$$\|x_k - \hat{x}\| \leqslant \left\|x_{k_N} - \hat{x}\right\| < \epsilon.$$

Since ϵ is arbitrary, we conclude that $x_k \to \hat{x}$ as $k\to\infty$.

2.10.3 Proof of Theorem 2.8

We need some preliminary results to establish Theorem 2.8.

Lemma 2.18. *Let $f:\mathbb{R}^n\to\mathbb{R}$ be convex and β-smooth. Then, for any $x,y\in\mathbb{R}^n$,*

$$\langle \nabla f(x) - \nabla f(y), x - y\rangle \geqslant \frac{1}{\beta}\|\nabla f(x) - \nabla f(y)\|^2.$$

Proof. See [157, Theorem 2.1.5]. □

Lemma 2.19. *Let $f:\mathbb{R}^n\to\mathbb{R}$ be convex and β-smooth. Then the operator $(I-\rho^{-1}\nabla f)$ is $(\beta/2\rho)$-averaged for $\rho > \beta/2$.*

Proof. Let $\theta = \beta/2\rho \in (0,1)$, and let $S = I - (2/\beta)\nabla f$. Then $I-\rho^{-1}\nabla f = (1-\theta)I + \theta S$. We are done if we can show that S is non-expansive. Indeed, for any $x,y\in\mathbb{R}^n$,

$$\begin{aligned}
\|S(x) - S(y)\|^2 &= \left\|(x-y) - \frac{2}{\beta}\big(\nabla f(x) - \nabla f(y)\big)\right\|^2 \\
&= \|x-y\|^2 + \frac{4}{\beta}\Big[\frac{1}{\beta}\|\nabla f(x) - \nabla f(y)\|^2 \\
&\qquad\qquad - \langle x-y, \nabla f(x) - \nabla f(y)\rangle\Big] \\
&\leqslant \|x-y\|^2,
\end{aligned}$$

where the last inequality follows from Lemma 2.18. □

Lemma 2.20. *If $T_1, T_2 : \mathbb{R}^n\to\mathbb{R}^n$ are averaged, then $T_1\circ T_2$ is averaged.*

Proof. See [114, Proposition 4.44]. □

We are now ready to prove Theorem 2.8. By Lemma 2.19, $(\mathrm{I}-\rho^{-1}\nabla f)$ is averaged if $\rho > \beta/2$. Since D is assumed to be averaged, it follows from Lemma 2.20 that $\mathrm{T}_{\mathrm{ISTA}}$ is averaged if $\rho > \beta/2$. Finally, since $\mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}}) \neq \varnothing$, Theorem 2.8 follows immediately from Lemma 2.6.

2.10.4 Proof of Theorem 2.9

We first show that the PnP-ADMM iterates $(v_k, x_k, z_k)_{k\geqslant 1}$ can be written in terms of a single-variable sequence $(u_k)_{k\geqslant 1}$.

Lemma 2.21. *Fix $\rho > 0$ and $x_0, z_0 \in \mathbb{R}^n$, and consider the sequence $(v_k, x_k, z_k)_{k\geqslant 1}$ generated by (2.12). Let $u_1 = x_1 + z_1$, and $u_k = \mathrm{T}_{\mathrm{ADMM}}(u_{k-1})$ for $k \geqslant 2$, where $\mathrm{T}_{\mathrm{ADMM}}$ is given by (2.13). Then for $k \geqslant 1$, we have $x_k = \mathrm{D}(u_k)$, $z_k = (\mathrm{I}-\mathrm{D})(u_k)$ and $v_{k+1} = \mathrm{Prox}_{\rho^{-1}f} \circ (2\mathrm{D}-\mathrm{I})(u_k)$.*

Proof. Define $\hat{y}_k = \mathrm{D}(u_k), \hat{z}_k = (\mathrm{I}-\mathrm{D})(u_k)$, and $\hat{x}_{k+1} = \mathrm{Prox}_{\rho^{-1}f} \circ (2\mathrm{D}-\mathrm{I})(u_k)$. We will inductively show that for $i \geqslant 1$,

$$x_i = \hat{y}_i,\ z_i = \hat{z}_i \text{ and } v_{i+1} = \hat{x}_{i+1}. \tag{2.31}$$

Base case: First, note that (2.31) holds for $i = 1$. Indeed, from (2.12b) and (2.12c), we have $x_1 = \mathrm{D}(v_1 + z_0)$ and $z_1 = z_0 + v_1 - x_1$. Since, by construction, $u_1 = x_1 + z_1$, we have $u_1 = z_0 + v_1$. Therefore, $x_1 = \mathrm{D}(v_1 + z_0) = \mathrm{D}(u_1) = \hat{y}_1$, and

$$z_1 = u_1 - x_1 = u_1 - \mathrm{D}(u_1) = (\mathrm{I}-\mathrm{D})(u_1) = \hat{z}_1.$$

Moreover, $x_1 - z_1 = (2\mathrm{D}-\mathrm{I})(u_1)$. Hence, from (2.12a), we get that $v_2 = \mathrm{Prox}_{\rho^{-1}f} \circ (2\mathrm{D}-\mathrm{I})(u_1) = \hat{x}_2$.

Induction: Assume that (2.31) holds for $1 \leqslant i \leqslant k-1$. We need to show that (2.31) holds for $i = k$. From the definition of $\mathrm{T}_{\mathrm{ADMM}}$ and the induction hypothesis, we have

$$\begin{aligned} u_k &= \mathrm{T}_{\mathrm{ADMM}}(u_{k-1}) \\ &= \mathrm{Prox}_{\rho^{-1}f} \circ (2\mathrm{D}-\mathrm{I})(u_{k-1}) + (\mathrm{I}-\mathrm{D})(u_{k-1}) \\ &= \hat{x}_k + \hat{z}_{k-1} = v_k + z_{k-1}. \end{aligned}$$

Using (2.12b), (2.12c) and the induction hypothesis, we see that

$$\hat{y}_k = \mathrm{D}(u_k) = \mathrm{D}(v_k + z_{k-1}) = x_k,$$

and

$$\hat{z}_k = (\mathrm{I}-\mathrm{D})(u_k) = u_k - \hat{y}_k = z_{k-1} + v_k - x_k = z_k.$$

Finally, note that $(2\mathrm{D}-\mathrm{I})(\boldsymbol{u}_k)=\boldsymbol{x}_k-\boldsymbol{z}_k$. Thus, from (2.12a) and the definition of $\hat{\boldsymbol{x}}_{k+1}$, we have

$$\hat{\boldsymbol{x}}_{k+1}=\mathrm{Prox}_{\rho^{-1}f}(\boldsymbol{x}_k-\boldsymbol{z}_k)=\boldsymbol{v}_{k+1}.$$

This completes the induction and the proof of Lemma 2.21. □

Next, we need a standard result about proximal operators.

Lemma 2.22. *Let $f:\mathbb{R}^n\to\overline{\mathbb{R}}$ be closed, proper and convex. Then Prox_f is $(1/2)$-averaged.*

Proof. See [114, Proposition 2.28, Proposition 4.4]. □

Finally, we need the following property of averaged operators.

Lemma 2.23. *If $\mathrm{T}:\mathbb{R}^n\to\mathbb{R}^n$ is θ-averaged, then T is $\bar{\theta}$-averaged for all $\bar{\theta}\in[\theta,1)$.*

Proof. Let $\mathrm{T}=(1-\theta)\mathrm{I}+\theta\mathrm{S}$, where $\theta\in(0,1)$ and S is non-expansive. Fix $\bar{\theta}\in[\theta,1)$ and define $\gamma=\theta/\bar{\theta}$. Then $\mathrm{T}=(1-\bar{\theta})\mathrm{I}+\bar{\theta}\bar{\mathrm{S}}$, where $\bar{\mathrm{S}}=(1-\gamma)\mathrm{I}+\gamma\mathrm{S}$. Since $\bar{\mathrm{S}}$ is non-expansive by triangle inequality and $\bar{\theta}\in[\theta,1)$ was arbitrary, it follows that T is averaged for all $\bar{\theta}\in[\theta,1)$. □

Using the above results, we can now establish Theorem 2.9. Note that:

- $2\mathrm{Prox}_{\rho^{-1}f}-\mathrm{I}$ is non-expansive since $\mathrm{Prox}_{\rho^{-1}f}$ is $(1/2)$-averaged (Lemma 2.22).
- $2\mathrm{D}-\mathrm{I}$ is non-expansive since D is $(1/2)$-averaged by Lemma 2.23.
- Thus, $(2\mathrm{Prox}_{\rho^{-1}f}-\mathrm{I})\circ(2\mathrm{D}-\mathrm{I})$ is non-expansive. Hence, $\mathrm{T}_{\mathrm{ADMM}}$ is $(1/2)$-averaged.
- Since $\mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})\neq\varnothing$, Lemma 2.6 implies that the sequence $(\boldsymbol{u}_k)_{k\geqslant 1}$ in Proposition 2.21 converges to some $\boldsymbol{u}^*\in\mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})$.
- Since D and $\mathrm{Prox}_{\rho^{-1}f}$ are continuous, we have

$$\begin{aligned}&\boldsymbol{v}_k\longrightarrow\mathrm{Prox}_{\rho^{-1}f}\circ(2\mathrm{D}-\mathrm{I})(\boldsymbol{u}^*),\\&\boldsymbol{x}_k\longrightarrow\mathrm{D}(\boldsymbol{u}^*)\ \text{ and }\ \boldsymbol{z}_k\longrightarrow(\mathrm{I}-\mathrm{D})(\boldsymbol{u}^*).\end{aligned}$$

This completes the proof of Theorem 2.9.

2.10.5 Proof of Theorem 2.11

(a) To prove (a), we need the following result from [158, Corollary 3.4]. Let $\mathrm{D}(\boldsymbol{x})=\mathbf{W}\boldsymbol{x}$ be a linear operator on $\mathbb{R}^n$ that is symmetric with respect to $\langle\cdot,\cdot\rangle$ and whose eigenvalues are in $[0,1]$. Then D is the proximal operator (with respect to $\langle\cdot,\cdot\rangle$) of a closed proper convex function. In the present context, the inner product $\langle\cdot,\cdot\rangle$ is given by (2.14). It is easily verified that the operator in question $\mathbf{W}$ is symmetric with respect to $\langle\cdot,\cdot\rangle$, i.e., $\langle\mathbf{W}\boldsymbol{x},\boldsymbol{y}\rangle=\langle\boldsymbol{x},\mathbf{W}\boldsymbol{y}\rangle$

for all $x, y \in \mathbb{R}^n$. Also, we already know that its eigenvalues are in $[0, 1]$. This establishes (a). We remark that a similar result is also used in [88], where $\mathbf{W}$ is assumed to be symmetric with respect to the standard inner product.

(b) From part (a), we have

$$\mathbf{W}x = \mathrm{Prox}_{h_{\mathrm{D}}}(x), \qquad (x \in \mathbb{R}^n) \tag{2.32}$$

where $h_{\mathrm{D}} : \mathbb{R}^n \to \overline{\mathbb{R}}$ is closed proper and convex. Hence, (b) follows from Lemma 2.22 and Lemma 2.23.

(c) From (2.32), $\mathrm{Prox}_{h_{\mathrm{D}}}$ maps $\mathbb{R}^n$ onto $\mathscr{R}(\mathbf{W})$. Moreover, since h_{D} is proper and from (2.32), h_{D} is real-valued on $\mathscr{R}(\mathbf{W})$. To prove that h_{D} restricted to $\mathscr{R}(\mathbf{W})$ is continuous, note that $\mathscr{R}(\mathbf{W})$ is a subspace of $\mathbb{R}^n$. If $\dim \mathscr{R}(\mathbf{W}) = 0$, then continuity follows trivially. Hence, assume that $\dim \mathscr{R}(\mathbf{W}) = p \geqslant 1$. Let $\varphi : \mathbb{R}^p \to \mathscr{R}(\mathbf{W})$ be a linear isomorphism, and define $h = h_{\mathrm{D}} \circ \varphi$. Since h_{D} is convex and φ is linear, h is convex. Since real-valued convex functions are continuous (see [114, Corollary 8.40]), h is continuous. Further, since φ^{-1} is continuous, it follows that $h_{\mathrm{D}}|_{\mathscr{R}(\mathbf{W})} = h \circ \varphi^{-1}$ is continuous.

(d) Since $\mathrm{D} = \mathrm{Prox}_{h_{\mathrm{D}}}$, we can replace the denoiser D by $\mathrm{Prox}_{h_{\mathrm{D}}}$ in (2.11) and (2.12b). Then, the updates in PnP-ISTA (resp. PnP-ADMM) are exactly that of ISTA (resp. ADMM) applied to optimization problem (2.15).

2.10.6 Proof of Theorem 2.12

To establish this theorem, we will require the concept of subdifferential of a convex function.

Definition 2.24. *Let $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ be proper and convex. The subdifferential of f at $x \in \mathbb{R}^n$, denoted by $\partial f(x)$, is defined to be the set of all $v \in \mathbb{R}^n$ such that*

$$f(y) \geqslant f(x) + \langle v, y - x \rangle \qquad (y \in \mathbb{R}^n).$$

As with the rest of the discussion, we assume $\langle \cdot, \cdot \rangle$ to be an arbitrary (but fixed) inner product on $\mathbb{R}^n$.

We need the following properties of subdifferential.

Lemma 2.25. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex and $g : \mathbb{R}^n \to \overline{\mathbb{R}}$ be closed proper and convex. Then*

1. *x^* minimizes $h(x) = g(x) + (\rho/2)\|x - \bar{x}\|^2$, where $\rho > 0$ and $\bar{x} \in \mathbb{R}^n$, if and only if $0 \in \partial g(x^*) + \rho(x^* - \bar{x})$.*
2. *x^* minimizes $f + g$ if and only if $0 \in \partial f(x^*) + \partial g(x^*)$. If f is differentiable, then $0 \in \nabla f(x^*) + \partial g(x^*)$.*

Proof. For (a), see [114, Th. 16.3 and Example 16.43]. For (b), see [114, Th. 16.3 and Corollary 16.48]. □

Proposition 2.26. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex and differentiable, and let $g : \mathbb{R}^n \to \overline{\mathbb{R}}$ be closed proper and convex. Then $\boldsymbol{x}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}})$ where $\mathrm{D} = \mathrm{Prox}_{\rho^{-1}g}$ if and only if $\boldsymbol{x}^*$ is a minimizer of $f + g$.*

Proof. Let $\boldsymbol{x}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}})$, where $\mathrm{T}_{\mathrm{ISTA}} = \mathrm{Prox}_{\rho^{-1}g} \circ (\mathrm{I} - \rho^{-1}\nabla f)$. Let $\boldsymbol{y}^* = (\mathrm{I} - \rho^{-1}\nabla f)(\boldsymbol{x}^*)$. Then $\nabla f(\boldsymbol{x}^*) = \rho(\boldsymbol{x}^* - \boldsymbol{y}^*)$. Moreover,

$$\boldsymbol{x}^* = \mathrm{T}_{\mathrm{ISTA}}(\boldsymbol{x}^*) = \operatorname*{argmin}_{\boldsymbol{y} \in \mathbb{R}^n} \; g(\boldsymbol{y}) + \frac{\rho}{2} \|\boldsymbol{y} - \boldsymbol{y}^*\|^2 .$$

Therefore, by Lemma 2.25, $0 \in \partial g(\boldsymbol{x}^*) + \rho(\boldsymbol{x}^* - \boldsymbol{y}^*)$. Thus,

$$0 \in \partial g(\boldsymbol{x}^*) + \rho(\boldsymbol{x}^* - \boldsymbol{y}^*) = \partial g(\boldsymbol{x}^*) + \nabla f(\boldsymbol{x}^*).$$

Using Lemma 2.25, we conclude that $\boldsymbol{x}^*$ is a minimizer of $f + g$. The other direction can be established by reversing the above steps. □

Using the above results, we can now arrive at Theorem 2.12. Note that since $\mathrm{D} \in \mathfrak{L}$, D is averaged by Proposition 2.11. Moreover, since (2.15) has a minimizer, $\mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}}) \neq \varnothing$ by Proposition 2.26. Therefore, from Theorem 2.8 and Proposition 2.26, we can conclude that $(\boldsymbol{x}_k)_{k \geqslant 0} \to \boldsymbol{x}^*$, where $\boldsymbol{x}^*$ is a minimizer of (2.15). In particular, since $\mathrm{D} = \mathrm{Prox}_{h_{\mathrm{D}}}$,

$$p^* = f(\boldsymbol{x}^*) + \rho h_{\mathrm{D}}(\boldsymbol{x}^*).$$

Now, $\boldsymbol{x}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}}) \subseteq \mathscr{R}(\mathbf{W})$ and $\boldsymbol{x}_k \in \mathscr{R}(\mathbf{W})$ for all k. On the other hand, f is continuous on $\mathbb{R}^n$ and h_{D} is continuous on $\mathscr{R}(\mathbf{W})$ by Theorem 2.11. Therefore, by continuity, we have

$$f(\boldsymbol{x}_k) + \rho h_{\mathrm{D}}(\boldsymbol{x}_k) \longrightarrow f(\boldsymbol{x}^*) + \rho h_{\mathrm{D}}(\boldsymbol{x}^*) = p^*.$$

This completes the proof of Theorem 2.12.

2.10.7 Proof of Theorem 2.13

We first prove a result similar to Proposition 2.26 that relates $\mathrm{fix}(T_{\mathrm{ADMM}})$ to the minimizers of (2.15).

Proposition 2.27. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex and $g : \mathbb{R}^n \to \overline{\mathbb{R}}$ be closed, proper and convex. Then $\boldsymbol{x}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})$ with $\mathrm{D} = \mathrm{Prox}_{\rho^{-1}g}$ if and only if $\mathrm{D}(\boldsymbol{x}^*)$ is a minimizer of $f + g$..*

Proof. Suppose $\boldsymbol{x}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})$, i.e.,

$$\boldsymbol{x}^* = \Big(2\mathrm{Prox}_{\rho^{-1}f} - \mathrm{I}\Big) \circ \Big(2\mathrm{Prox}_{\rho^{-1}g} - \mathrm{I}\Big)(\boldsymbol{x}^*).$$

Let $\boldsymbol{y}^* = \left(2\mathrm{Prox}_{\rho^{-1}g} - \mathrm{I}\right)(\boldsymbol{x}^*)$. Note that

$$\mathrm{Prox}_{\rho^{-1}g}(\boldsymbol{x}^*) = \frac{\boldsymbol{x}^* + \boldsymbol{y}^*}{2} = \mathrm{Prox}_{\rho^{-1}f}(\boldsymbol{y}^*). \tag{2.33}$$

From Lemma 2.25, we have

$$\begin{aligned} 0 &\in \partial g\Big(\frac{\boldsymbol{x}^* + \boldsymbol{y}^*}{2}\Big) + \frac{\rho}{2}(\boldsymbol{y}^* - \boldsymbol{x}^*), \text{ and} \\ 0 &\in \partial f\Big(\frac{\boldsymbol{x}^* + \boldsymbol{y}^*}{2}\Big) + \frac{\rho}{2}(\boldsymbol{x}^* - \boldsymbol{y}^*). \end{aligned}$$

Adding these inclusions and using (2.33), we obtain

$$0 \in \partial f\left(\mathrm{Prox}_{\rho^{-1}g}(\boldsymbol{x}^*)\right) + \partial g\left(\mathrm{Prox}_{\rho^{-1}g}(\boldsymbol{x}^*)\right).$$

Since $\mathrm{D} = \mathrm{Prox}_{\rho^{-1}g}$, we conclude from Lemma 2.25 that $\mathrm{D}(\boldsymbol{x}^*)$ is a minimizer of $f + g$.

Conversely, let $\boldsymbol{y}^* = \mathrm{Prox}_{\rho^{-1}g}(\boldsymbol{x}^*)$. From Lemma 2.25, we have $0 \in \partial g(\boldsymbol{y}^*) + \rho(\boldsymbol{y}^* - \boldsymbol{x}^*)$. Moreover, since $\boldsymbol{y}^* = \mathrm{D}(\boldsymbol{x}^*)$, $\boldsymbol{y}^*$ is a minimizer of $f + g$. Using Lemma 2.25 once more, we have $0 \in \partial f(\boldsymbol{y}^*) + \partial g(\boldsymbol{y}^*)$. Combining these, we obtain $0 \in \partial f(\boldsymbol{y}^*) + \rho(\boldsymbol{x}^* - \boldsymbol{y}^*)$, i.e., $\boldsymbol{y}^* = \mathrm{Prox}_{\rho^{-1}f}(2\boldsymbol{y}^* - \boldsymbol{x}^*)$, or equivalently,

$$\boldsymbol{x}^* = \left(2\mathrm{Prox}_{\rho^{-1}f} - \mathrm{I}\right)(2\boldsymbol{y}^* - \boldsymbol{x}^*). \tag{2.34}$$

However, $2\boldsymbol{y}^* - \boldsymbol{x}^* = \left(2\mathrm{Prox}_{\rho^{-1}g} - \mathrm{I}\right)(\boldsymbol{x}^*)$. Therefore, from (2.34), we have

$$\boldsymbol{x}^* = \left(2\mathrm{Prox}_{\rho^{-1}f} - \mathrm{I}\right) \circ \left(2\mathrm{Prox}_{\rho^{-1}g} - \mathrm{I}\right)(\boldsymbol{x}^*),$$

i.e., $\boldsymbol{x}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})$ with $\mathrm{D} = \mathrm{Prox}_{\rho^{-1}g}$. □

We can now establish Theorem 2.13. Since $\mathrm{D} \in \mathfrak{L}$, D is (1/2)-averaged by Theorem 2.11. Since (2.15) has a minimizer, $\mathrm{fix}(T_{\mathrm{ADMM}}) \neq \varnothing$ by Proposition 2.27. Using Theorem 2.9, we can conclude that there exists $\boldsymbol{u}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})$ such that $(\boldsymbol{x}_k)_{k\geqslant 0}$ converges to $\boldsymbol{y}^* = \mathrm{Prox}_{h_{\mathrm{D}}}(\boldsymbol{u}^*)$. Hence, by Proposition 2.27, $\boldsymbol{y}^*$ is a minimizer of (2.15), i.e., $f(\boldsymbol{y}^*) + \rho h_{\mathrm{D}}(\boldsymbol{y}^*) = p^*$. Finally, note that $\boldsymbol{y}^* = \mathrm{D}(\boldsymbol{u}^*) \in \mathscr{R}(\mathbf{W})$ and $(\boldsymbol{x}_k) \subseteq \mathscr{R}(\mathbf{W})$ for all k. Since h_{D} is continuous on $\mathscr{R}(\mathbf{W})$ by Theorem 2.11 and f is continuous, $f(\boldsymbol{x}_k) + \rho h_{\mathrm{D}}(\boldsymbol{x}_k) \longrightarrow f(\boldsymbol{y}^*) + \rho h_{\mathrm{D}}(\boldsymbol{y}^*) = p^*$. This completes the proof of Theorem 2.13.

2.10.8 Proof of Proposition 2.15

Since the kernel ϕ is symmetric and positive definite, it follows from Definition 2.14 that the kernel matrix $\mathbf{K}$ in (2.17) is symmetric and positive semidefinite. Thus, the matrix $\mathbf{W} = \mathbf{D}^{-1}\mathbf{K}$

is nonnegative and stochastic. Therefore, $\mathbf{W}$ has a complete set of eigenvectors with eigenvalues in $[0,1]$ [132], which implies that $\mathrm{D} \in \mathfrak{L}$.

For the second part, we apply Theorem 2.11b). This tells us that D is θ-averaged on $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_{\mathbf{D}})$ for every $\theta \in [1/2, 1)$, where $\langle\cdot,\cdot\rangle_{\mathbf{D}}$ is given by (2.14). In particular, let $\mathbf{B} = \mathbf{D}^{-\frac{1}{2}}\mathbf{K}\mathbf{D}^{-\frac{1}{2}}$ and $\mathbf{B} = \mathbf{U}\Sigma\mathbf{U}^\top$ be its eigendecomposition. We can then write $\mathbf{W} = \mathbf{D}^{-\frac{1}{2}}\mathbf{B}\mathbf{D}^{\frac{1}{2}} = \mathbf{V}\Sigma\mathbf{V}^{-1}$, where $\mathbf{V} = \mathbf{D}^{-\frac{1}{2}}\mathbf{U}$ is an eigen matrix of $\mathbf{W}$. Using this eigen matrix, $\langle\cdot,\cdot\rangle_{\mathbf{D}}$ is given by

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathbf{D}} = \left\langle \mathbf{V}^{-1}\boldsymbol{x}, \mathbf{V}^{-1}\boldsymbol{y} \right\rangle_2 = \boldsymbol{x}^\top \mathbf{D} \boldsymbol{y},$$

where we have used the fact that $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$. This completes the proof of Proposition 2.15.

2.10.9 Proof of Theorem 2.16

We will first establish that $\mathbf{K}$ is positive definite using the following result [159, Chapter 13, Lemma 6]. In this part, we use ι to denote $\sqrt{-1}$.

Proposition 2.28. *Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ be distinct points in $\mathbb{R}^n$ and let $c_1, \ldots, c_n \in \mathbb{R}$ be such that not all are zero. Then the function $h : \mathbb{R}^n \to \mathbb{C}$ defined as*

$$h(\boldsymbol{y}) = \sum_{i=1}^{n} c_i \exp\left(\iota \langle \boldsymbol{y}, \boldsymbol{x}_k \rangle_2\right)$$

is non-zero almost everywhere.

Since $\mathbf{K}$ is symmetric, we need to show that $\boldsymbol{c}^\top\mathbf{K}\boldsymbol{c} > 0$ for all non-zero $\boldsymbol{c} \in \mathbb{R}^N$. Let c_s denote the element of $\boldsymbol{c}$ corresponding to the spatial location $\boldsymbol{s} \in \Omega \subset \mathbb{Z}^2$, where Ω is the support of the image. From (2.20), we have

$$\begin{aligned}
\boldsymbol{c}^\top\mathbf{K}\boldsymbol{c} &= \sum_{s\in\Omega}\sum_{t\in\Omega} c_s c_t \, \Lambda(\boldsymbol{s}-\boldsymbol{t})\mu(\mathbf{P}_s - \mathbf{P}_t) \\
&= \sum_{s\in\Omega}\sum_{t\in\Omega} \left\{ c_s c_t \int_{\mathbb{R}^2} \hat{\Lambda}(\boldsymbol{\omega}_1) e^{\iota\boldsymbol{\omega}_1^\top(\boldsymbol{s}-\boldsymbol{t})} d\boldsymbol{\omega}_1 \right. \\
&\qquad \left. \times \int_{\mathbb{R}^P} \hat{\mu}(\boldsymbol{\omega}_2) e^{\iota\boldsymbol{\omega}_2^\top(\mathbf{P}_s - \mathbf{P}_t)} d\boldsymbol{\omega}_2 \right\}
\end{aligned}$$

where $\hat{\Lambda}$ and $\hat{\mu}$ are the Fourier transforms of Λ and μ, and P is the patch size. Let $\boldsymbol{\omega} = (\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) \in \mathbb{R}^{2+P}$ and $\zeta_s = (\boldsymbol{s}, \mathbf{P}_s) \in \mathbb{R}^{2+P}$. Switching the sums and the integrals, and using Fubini's theorem, we get

$$\boldsymbol{c}^\top\mathbf{K}\boldsymbol{c} = \int_{\mathbb{R}^{2+P}} \hat{\Lambda}(\boldsymbol{\omega}_1)\hat{\mu}(\boldsymbol{\omega}_2)|h(\boldsymbol{\omega})|^2 d\boldsymbol{\omega}, \tag{2.35}$$

where $h(\boldsymbol{\omega}) = \sum_{s\in\Omega} c_s \exp(\iota\boldsymbol{\omega}^\top\zeta_s)$. Since the points ζ_s are distinct for all $\boldsymbol{s} \in \Omega$, we can

conclude from Proposition 2.28 that $h(\boldsymbol{\omega})$ is non-zero almost everywhere on $\mathbb{R}^{2+P}$. Moreover, since $\hat{\Lambda}$ is positive almost everywhere on $\mathbb{R}^2$ and $\hat{\mu}$ is positive everywhere on $\mathbb{R}^P$, the function $\boldsymbol{\omega} \mapsto \hat{\Lambda}(\boldsymbol{\omega}_1)\hat{\mu}(\boldsymbol{\omega}_2)$ is positive almost everywhere on $\mathbb{R}^{2+P}$. Hence, the integral (2.35) is positive and $\mathbf{K}$ is positive definite.

Next, note that since $\mathbf{K}$ is invertible, we can define

$$h_{\mathrm{D}}(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^{\top}\mathbf{D}(\mathbf{K}^{-1}\mathbf{D}-\mathbf{I})\boldsymbol{x}. \tag{2.36}$$

Now, $\mathbf{D}(\mathbf{K}^{-1}\mathbf{D}-\mathbf{I}) = \mathbf{D}\mathbf{K}^{-1}\mathbf{D}-\mathbf{D}$ is symmetric and we can verify that its eigenvalues are nonnegative. Hence, (2.36) is convex. Furthermore, note that $\mathbf{W} = \mathbf{D}^{-1}\mathbf{K}$ and $h_{\mathrm{D}}(\boldsymbol{x}) = (1/2)\langle \boldsymbol{x}, (\mathbf{K}^{-1}\mathbf{D}-\mathbf{I})\boldsymbol{x}\rangle_{\mathbf{D}}$, where $\langle \boldsymbol{x}, \boldsymbol{x}\rangle_{\mathbf{D}} = \boldsymbol{x}^{\top}\mathbf{D}\boldsymbol{x}$ is the inner product in Proposition 2.16. Claim is $\mathbf{W}\boldsymbol{x} = \mathrm{Prox}_{h_{\mathrm{D}}}(\boldsymbol{x})$, i.e.,

$$\mathbf{W}\boldsymbol{x} = \underset{\boldsymbol{y}\in\mathbb{R}^n}{\operatorname{argmin}} \ \frac{1}{2}\left\langle \boldsymbol{y}, (\mathbf{K}^{-1}\mathbf{D}-\mathbf{I})\boldsymbol{y}\right\rangle_{\mathbf{D}} + \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{x}\|_{\mathbf{D}}^2, \tag{2.37}$$

where $\|\cdot\|_{\mathbf{D}}$ is induced by $\langle\cdot,\cdot\rangle_{\mathbf{D}}$. This is because the derivative of the objective in (2.37) is zero at $\boldsymbol{y} = \mathbf{W}\boldsymbol{x}$.

Proof of proposition 2.28

We rewrite the proof for proposition 2.28 from [159] in simple notations.

Proposition 2.29. *Let $\lambda_1, \ldots, \lambda_n$ be distinct points in $\mathbb{R}$ and let $c_1, \ldots, c_n \in \mathbb{R}$ be complex numbers such that not all are zero. Let $\iota = \sqrt{-1}$. Then the function $h : \mathbb{R} \to \mathbb{C}$ defined as*

$$h(x) = \sum_{i=1}^{n} c_k \exp(\iota\lambda_k x)$$

is non-zero.

Proof. It suffices to show that the set of complex exponentials $\left\{\exp(\iota\lambda_k x) : k \in \{1, 2, \ldots n\}\right\}$ are independent. Refer to [159, Chapter 13, Lemma 1] for the proof of this statement . □

Proposition 2.30. *Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ be nonzero points in $\mathbb{R}^s$, where $s \geqslant 2$. If $\Pi : \mathbb{R}^s \to \mathbb{R}^{(s-1)}$ projects vectors in $\mathbb{R}^s$ to $\mathbb{R}^{s-1}$ by deleting the last coordinate of points, then there exists an orthonormal matrix $\mathbf{U} \in \mathbb{R}^{s\times s}$, such that $\Pi(\mathbf{U}\boldsymbol{x}_i)$ are nonzero in $\mathbb{R}^{s-1}$.*

Proof. See [159, Chapter 13, Lemma 5]. It suffices to show that that $\mathbf{U}\boldsymbol{x}_i \notin span(\boldsymbol{v})$, where $\boldsymbol{v} \in \mathbb{R}^s$ is defined as $(0, 0, \ldots 0, 1)$, all zeros with 1 only in the last coordinate. We require the following claim, □

Proposition 2.31. *Let $(\mathbf{X}, \mathbf{A}, \mu)$ and $(\mathbf{Y}, \mathbf{B}, \nu)$ be σ-finite measure spaces. Then $(\mathbf{X}\times\mathbf{Y}, \mathbf{A}\otimes\mathbf{B}, \mu\times\nu)$ is the product measure space. Let f be a measurable function on $\mathbf{X}\times\mathbf{Y}$. If for all $\boldsymbol{x} \in \mathbf{X}$, $f(\boldsymbol{x}, \boldsymbol{y}) \neq 0$ almost everywhere on $\mathbf{Y}$, then $f(\boldsymbol{x}, \boldsymbol{y}) \neq 0$ almost everywhere on $\mathbf{X}\times\mathbf{Y}$.*

Proof. See [159, Chapter 13, Lemma 7] □

Proposition 2.28 is proved by induction on the dimension. The statement holds true for $s = 1$ by Proposition 2.29. Suppose the statement holds true for dimensions $1, 2, \ldots s-1$, we will prove that statement will hold for dimension s.

Let h be as in hypothesis and $\Pi : \mathbb{R}^s \to \mathbb{R}^{(s-1)}$ be as in Proposition 2.30. By applying Proposition 2.30 to nonzero differences $\boldsymbol{x}_i - \boldsymbol{x}_j$, there exists orthogonal matrix $\mathbf{U} \in \mathbb{R}^{s\times s}$ such that $\Pi(\mathbf{U}\boldsymbol{x}_i)$ are distinct. In particular, if we write each point $\mathbf{U}\boldsymbol{x}_k = (\boldsymbol{v}_k, \lambda_k)$, where $\boldsymbol{v}_k \in \mathbb{R}^{s-1}$ and $\lambda_k \in \mathbb{R}$, then the points $\boldsymbol{v}_k$ are distinct in $\mathbb{R}^{s-1}$. Note that it suffices to show that $g(\boldsymbol{x}) = h(\mathbf{U}^\top \boldsymbol{x})$ is zero almost everywhere on $\mathbb{R}^s$. Write $\boldsymbol{x} = (\boldsymbol{y}, \xi)$ with $\boldsymbol{y} \in \mathbb{R}^{s-1}$ and $\xi \in \mathbb{R}$. Then we have,

$$\begin{aligned} g(\boldsymbol{x}) = h(\mathbf{U}^\top \boldsymbol{x}) &= \sum_{i=1}^{n} c_k \exp(\imath \boldsymbol{x}_k^\top \mathbf{U}^\top \boldsymbol{x}) = \sum_{i=1}^{n} c_k \exp\left(\imath (\mathbf{U}\boldsymbol{x}_k)^\top \boldsymbol{x}\right) \\ &= \sum_{i=1}^{n} c_k \exp(\imath {\boldsymbol{v}_k}^\top \boldsymbol{y} + \imath \lambda_k \xi) = \sum_{i=1}^{n} c_k \exp(\imath \lambda_k \xi) \exp(\imath {\boldsymbol{v}_k}^\top \boldsymbol{y}). \end{aligned}$$

We can interpret function g on $\mathbb{R}^{s-1} \times \mathbb{R}$ and hence,

$$g(\boldsymbol{y}, \xi) = \sum_{i=1}^{n} c_k \exp(\imath \lambda_k \xi) \exp(\imath {\boldsymbol{v}_k}^\top \boldsymbol{y}).$$

Fix $\xi \in \mathbb{R}$. Since c_k are not all zero by given hypothesis, $c_k \exp(\imath \lambda_k \xi)$ are not all zero. Then $g_\xi(\boldsymbol{y})$ is different from zero almost everywhere on $\mathbb{R}^{s-1}$ from induction hypothesis assumption. Then by Proposition 2.31, we get $g(\boldsymbol{y}, \xi) \neq 0$ almost everywhere in $(\boldsymbol{y}, \xi)$. This implies, $g(\boldsymbol{x}) \neq 0$ almost everywhere on $\mathbb{R}^s$.

PnP Regularization using Linear Solvers

Abstract. *In this chapter, we focus on the special class of linear denoisers which was introduced in Chapter 2. In particular, we show that these linear denoisers are indeed a proximal map of a convex regularizer which is quadratic and can accurately model natural images, which can be optimized efficiently along with the loss function. The quadratic regularizer proposed in this chapter is shown to have competitive reconstruction capabilities as compared to state-of-the-art regularizers. The novelty of the regularizer is that, unlike classical quadratic regularizers, the quadratic form is derived from the observed data. Since the regularizer is quadratic, for linear inverse problems such as superresolution, deblurring, inpainting, etc., we can reduce the optimization problem to solving a linear system. In particular, we show that using iterative Krylov solvers, we can converge to the solution in just a few iterations, where each iteration requires an application of the forward operator and a linear denoiser. The surprising finding is that we are able to get close to deep learning methods in terms of reconstruction quality. To the best of our knowledge, the possibility of achieving near state-of-the-art performance using a linear solver is novel.*

3.1 Introduction

In this chapter, we focus only on linear inverse problems. Recall from Chapter 1 that in linear inverse problems in Table 1.1, the measurements $\boldsymbol{y} \in \mathbb{R}^m$ are obtained by the degradation process $\boldsymbol{y} = \mathbf{A}\boldsymbol{x}_0 + \boldsymbol{\eta}$, where $\boldsymbol{x}_0 \in \mathbb{R}^n$ is ground-truth image, $\mathbf{A} \in \mathbb{R}^{m \times n}$ is the forward matrix and $\boldsymbol{\eta}$ is Gaussian noise. Since the data-fidelity term for linear inverse problems are convex, convergence guarantees for PnP algorithms derived in the previous chapter hold. Importantly, in Theorems 2.12 and 2.13 in Chapter 2, we proved that the reconstruction obtained using PnP-ISTA and PnP-ADMM with a special class of linear denoisers is equivalent to solving an optimization problem of the form (1.6). The class of linear denoisers under consideration was characterized as follows.

Definition 3.1. *Let $\mathfrak{L}$ denote the class of linear denoisers $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$ on $\mathbb{R}^n$ such that $\mathbf{W}$ is diagonalizable and its eigenvalues are in $[0, 1]$.*

More precisely, it was shown in Theorem 2.11 that if the linear operator D is diagonalizable with eigenvalues in $[0, 1]$, then we can represent D as the proximal operator of a convex

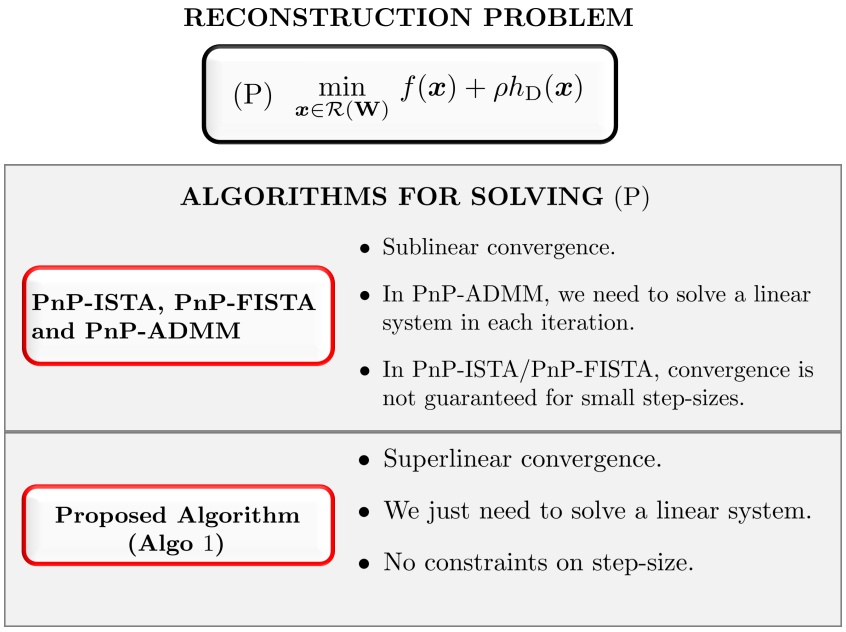


Figure 3.1: Relation of the proposed algorithm with PnP algorithms in Chapter 2 that use a linear denoiser $\mathbf{W}$. The reconstruction returned by such algorithms is a minimizer of a convex objective function $f + h_{\mathrm{D}}$. Instead of iteratively computing the minimizer (as done in PnP algorithms), we propose to solve the first-order optimality condition for the problem, which can be reduced to solving a linear system if the loss f is quadratic. See text for more details.

regularizer h_{D}. Two important points in this regard are (i) the effective domain of h_{D} is a subspace of $\mathbb{R}^n$ and (ii) the norm in the definition of the proximal operator is a weighted ℓ_2 norm and not the standard ℓ_2 norm (the exact description is provided in Section 3.3.1). The implication of this finding is that performing the PnP updates in (2.11) and (2.12) with a diagonalizable linear denoiser with eigenvalues in $[0, 1]$ amounts to minimizing an objective of the form $f + h_{\mathrm{D}}$. As will be made precise later, the regularizer h_{D} is derived from a linear denoiser which in turn is computed from some surrogate of the ground-truth image, derived from measurements.

The work in this chapter is best motivated using an analogy with least-squares regression, say, we wish to minimize $||\mathbf{A}\boldsymbol{x} - \boldsymbol{b}||^2$. Among other things, we can compute the minimizer using either an iterative algorithm such as the gradient descent method or via the solution of the normal equation $\mathbf{A}^\top\mathbf{A}\boldsymbol{x} = \mathbf{A}^\top\boldsymbol{b}$ [160]. Recall that the normal equation is the first-order optimality condition for the optimization problem at hand. In particular, if $\mathbf{A}$ has full column rank, then we can solve the normal equation using the conjugate gradient, which has superior convergence than the sublinearly-convergent gradient descent [161]. Similarly, our main idea is that instead of the iterative minimization in (2.11) and (2.12), which in effect is known to minimize $f + h_{\mathrm{D}}$, we wish to directly work with the first-order optimality condition for the optimization problem and come up with an efficient algorithm (see Fig. 3.1). The technical novelty in this regard are:

- We prove that the minimization of $f + h_{\mathrm{D}}$ is well-posed, i.e., it is guaranteed to have a minimizer $\boldsymbol{x}^* \in \mathbb{R}^n$ (Proposition 3.5). Moreover, we prove that $\boldsymbol{x}^*$ is unique for the imaging problems we are interested in (Proposition 3.6).
- We show that $\boldsymbol{x}^*$ in question can be computed by solving a linear system $\mathbf{F}\boldsymbol{x} = \boldsymbol{b}$. In particular, we prove that $\mathbf{F}$ is nonsingular for some reconstruction problems.
- On the computational side, we show how $\boldsymbol{x}^*$ can be accurately computed in just few iterations using superlinearly convergent solvers such as GMRES [162], LGMRES [163], GCROT [164] and Broyden [165, 166], where each iteration requires us to apply the forward operator $\mathbf{A}$ and $\mathbf{A}^\top$ and a kernel denoiser. Stated differently, we can obtain the same reconstruction as PnP-ISTA and PnP-ADMM, but unlike PnP-ISTA and PnP-ADMM which are known to converge sublinearly [40, 167], we can achieve superlinear convergence (refer Fig. 3.1 for a brief comparison of the proposed algorithm with PnP solvers). The speedup is apparent in practice as well.

Surprisingly, for superresolution, deblurring, and inpainting, we are able to get close to state-of-the-art methods in terms of reconstruction quality. This demonstrates that as a natural image prior, h_{D} is more effective than classical wavelet and total-variation priors.

3.2 Background

In this section, we will collect the background details required to explain our novel contributions presented in this Chapter.

3.2.1 Kernel Filter

Recall from Chapter 2 that a kernel denoiser D is abstractly a linear transform $\mathbf{W} \in \mathbb{R}^{n\times n}$ that takes a noisy vectorized image $\boldsymbol{x}$ and returns the denoised output $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$. The essential point is that $\mathbf{W}$ is defined using a symmetric positive definite kernel $\phi : \mathscr{F} \times \mathscr{F} \to \mathbb{R}_+$, where the feature space $\mathscr{F} \subset \mathbb{R}^k$ is stipulated using a guide image. The guide image is usually the noisy input for image denoising [68]. However, in PnP regularization, the guide image used to construct $\mathbf{W}$ is a surrogate of the ground-truth image $\boldsymbol{x}_0$ which is computed from the measurement $\boldsymbol{y}$ [79, 168].

In NLM denoiser, the feature vector $\boldsymbol{u} \in \mathbf{X}$ consists of the position of a pixel and the intensity values in a patch around the pixel, and $\phi(\boldsymbol{u}, \boldsymbol{u}') = G(\boldsymbol{u} - \boldsymbol{u}')$ where G is a multivariate Gaussian [134, 133]. The similarity between pixels indexed at i and j in the input image $\boldsymbol{x}$ is measured using $\phi(\boldsymbol{u}_i, \boldsymbol{u}_j)$, where $\boldsymbol{u}_i$ is the feature vector for the pixel indexed at $i \in [1, n]$. This is encoded into a kernel matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$ defined as $\mathbf{K}_{i,j} = \phi(\boldsymbol{u}_i, \boldsymbol{u}_j), 1 \leqslant i, j \leqslant n$. The kernel filter $\mathbf{W}$ is obtained by row-normalizing K:

$$\mathbf{W} = \mathbf{D}^{-1}\mathbf{K},$$

where $\mathbf{D}$ is a diagonal matrix and $\mathbf{D}_{i,i} = \sum_{j=1}^{n} \mathbf{K}_{i,j}, 1 \leqslant i \leqslant n$. The kernel denoiser is $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$. By construction, both $\mathbf{K}$ and $\mathbf{D}$ are symmetric; K is positive semidefinite and $\mathbf{D}$ is positive definite [127].

3.2.2 Kernel Denoiser as a Proximal Operator

Kernel filters, and in particular NLM-type denoisers, have been shown to be very effective for PnP regularization [79, 128, 131, 169]. In this section, we discuss existing results on the optimality of the PnP reconstruction obtained using a kernel denoiser. Recall that in PnP, we replace the proximal operator of a regularizer in (1.10) by a denoiser D in (1.20). The question is can we express a given denoiser as a proximal operator of some convex regularizer? The convergence of the PnP iterates and the optimality of the final reconstruction would immediately be settled if we can answer this in the affirmative. The resolution of this question remains open for complex denoisers such as BM3D and DnCNN [78]. However, it is shown in Chapter 2 that for a kernel denoiser D, there exists a convex regularizer h_{D} such that $\mathrm{D}(\boldsymbol{x}) = \mathrm{prox}_{h_{\mathrm{D}}}$. Moreover in [168], a form for the regularizer is proposed. We need some notations and definitions to explain our technical results. Note that some of them are recalled from the previous chapter for ease of explanation.

Note that, since $\mathbf{D}$ is positive definite, we defined the following inner product and norm on $\mathbb{R}^n$ in the Chapter 2:

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathrm{D}} = \boldsymbol{x}^{\top}\mathbf{D}\boldsymbol{y} \quad \text{and} \quad \|\boldsymbol{x}\|_{\mathrm{D}} = \langle \boldsymbol{x}, \boldsymbol{x} \rangle_{\mathrm{D}}^{1/2}. \tag{3.1}$$

The latter can be viewed as a weighted ℓ_2 norm. We also associate a proximal operator with this norm in Definition 2.3. For a convex function g, we will call Prox_g the scaled proximal operator of g as it is the norm induced by the denoiser, in particular $\mathbf{D}$. The term "scaled" is also used to emphasize that the weighted ℓ_2 norm is used in (2.3). It is well-known that the objective in (2.3) has a unique minimizer for the standard ℓ_2 norm and that Prox_g is well-defined [114]; it is not difficult to verify that this is also true for the weighted norm. Recall that in Chapter 2, we showed a kernel denoiser is a scaled proximal operator of a convex regularizer. This observation was used to establish objective and iterate convergence. We recall the exact formula for this regularizer in [168] which will play a central role in the rest of the discussion.

Let $\mathbf{W} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^{-1}$ be the eigendecomposition of $\mathbf{W}$, where the columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{W}$ and $\boldsymbol{\Lambda}$ is a diagonal matrix whose first r diagonal entries are in $(0, 1]$ and the remaining entries are 0. We collect the r positive eigenvalues in a diagonal matrix $\boldsymbol{\Lambda}_r$ and the corresponding eigenvectors in $\mathbf{U} \in \mathbb{R}^{n \times r}$. Then we can write $\mathbf{W} = \mathbf{U}\boldsymbol{\Lambda}_r\mathbf{U}^{\dagger}$, where $\mathbf{U}^{\dagger} \in \mathbb{R}^{r \times n}$ consists of the first r rows of $\mathbf{V}^{-1}$. The authors in [168] showed that the kernel

denoiser $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$ is a scaled proximal operator of the extended-valued convex function

$$h_{\mathrm{D}}(\boldsymbol{x}) = \begin{cases} \frac{1}{2}\boldsymbol{x}^{\top}\mathbf{U}^{\dagger\top}(\mathbf{\Lambda}_r^{-1} - \mathrm{I})\mathbf{U}^{\dagger}\boldsymbol{x}, & \text{for } \boldsymbol{x} \in \mathscr{R}(\mathbf{W}), \\ \infty, & \text{otherwise,} \end{cases} \tag{3.2}$$

where $\mathscr{R}(\mathbf{W})$ is the range space of $\mathbf{W}$. Note that h_{D} is completely determined by the kernel denoiser $\mathbf{W}$. Its domain is restricted to $\mathscr{R}(\mathbf{W})$ and this in effect forces the reconstruction to be in $\mathscr{R}(\mathbf{W})$.

3.3 Kernel Regularization

3.3.1 PnP Regularization using Kernel Denoiser

We first propose an alternate formulation for the regularizer in (3.2) and show that the scaled proximal operator of (3.2) is indeed $\mathbf{W}$, albeit using our alternate formulation. Apart from keeping the account self-contained, our formulation also makes the proofs simpler and helps analyze the image reconstruction problem (1.6) with $g = h_{\mathrm{D}}$.

Note that though $\mathbf{W}$ is not symmetric, it can be written as

$$\mathbf{W} = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{1/2}, \quad \mathbf{S} := \mathbf{D}^{-1/2}\mathrm{K}\mathbf{D}^{-1/2}.$$

Thus, $\mathbf{W}$ is similar to the symmetric matrix S and is hence diagonalizable with real eigenvalues. More specifically, if $\mathbf{S} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{\top}$ is an eigendecomposition of S, then we can write

$$\mathbf{W} = \mathbf{M}\mathbf{\Lambda}\mathbf{M}^{-1}, \quad \mathbf{M} := \mathbf{D}^{-1/2}\mathbf{Q}.$$

Define $\mathbf{W}^{\dagger} = \mathbf{M}\mathbf{\Lambda}^{\dagger}\mathbf{M}^{-1}$, where $\mathbf{\Lambda}^{\dagger}$ is diagonal and $\mathbf{\Lambda}^{\dagger}_{ii} := \mathbf{\Lambda}^{-1}_{ii}$ if $\mathbf{\Lambda}_{ii} > 0$ and $= 0$ otherwise. We remark that $\mathbf{W}^{\dagger}$ is the generalized inverse of $\mathbf{W}$ in that $\mathbf{W} = \mathbf{W}\mathbf{W}^{\dagger}\mathbf{W}$ [170].

Definition 3.2. *Let* D *be a kernel denoiser. Define the function* $h_{\mathrm{D}} : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ *to be*

$$h_{\mathrm{D}}(\boldsymbol{x}) = \begin{cases} \frac{1}{2}\boldsymbol{x}^{\top}\mathbf{D}(\mathbf{I} - \mathbf{W})\mathbf{W}^{\dagger}\boldsymbol{x}, & \text{for } \boldsymbol{x} \in \mathscr{R}(\mathbf{W}), \\ \infty, & \text{otherwise.} \end{cases} \tag{3.3}$$

We will call h_{D} *the kernel regularizer associated with kernel denoiser* $\mathrm{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$.

The extended-valued function h_{D} is quadratic on its domain. Moreover, we have the following properties.

Proposition 3.3. h_{D} *is nonnegative, closed (epigraph is closed in* $\mathbb{R}^{n+1}$*), proper (domain is nonempty), and convex.*

In particular, it follows from Proposition 3.3 that $\text{Prox}_{h_{\text{D}}}$ is well-defined. This brings us to the main result, namely, the characterization of a kernel denoiser as a proximal operator.

Theorem 3.4. *The kernel denoiser* D *is a scaled proximal operator of* h_{D}*, i.e.,*

$$\forall\ \boldsymbol{x} \in \mathbb{R}^n : \quad \text{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x} = \text{Prox}_{h_{\text{D}}}(\boldsymbol{x}),$$

with respect to inner product in (3.1)*.*

We now relate the above results to the PnP updates in (2.11) and (2.12) where the denoiser is $\text{D}(\boldsymbol{x}) = \mathbf{W}\boldsymbol{x}$. In this case, it follows from Theorem 3.4 that we can write (1.20) as

$$\boldsymbol{x}_{k+1} = \text{Prox}_{h_{\text{D}}}\big(\boldsymbol{x}_k - \rho^{-1}\nabla f(\boldsymbol{x}_k)\big). \tag{3.4}$$

This looks like standard ISTA [40, 114], except that we have the scaled proximal operator instead of the standard proximal operator. Thus, if we replace ∇f (gradient of f w.r.t. ℓ_2 norm) in (3.4) with $\mathbf{D}^{-1}\nabla f$ (gradient of f w.r.t. weighted-ℓ_2 norm), then PnP updates amount to minimizing $f + \rho h_{\text{D}}$ as shown in Chapter 2. Similarly, we can write the PnP-ADMM updates as

$$\begin{aligned}
\boldsymbol{v}_{k+1} &= \underset{\boldsymbol{x}}{\text{argmin}}\ f(\boldsymbol{x}) + \frac{1}{2\rho}||\boldsymbol{x} - (\boldsymbol{x}_k - \boldsymbol{z}_k)||_{\text{D}}^2, \\
\boldsymbol{x}_{k+1} &= \text{Prox}_{h_{\text{D}}}(\boldsymbol{v}_{k+1} + \boldsymbol{z}_k), \\
\boldsymbol{z}_{k+1} &= \boldsymbol{z}_k + (\boldsymbol{v}_{k+1} - \boldsymbol{x}_{k+1}),
\end{aligned} \tag{3.5}$$

Again, this looks like standard ADMM except that the weighted ℓ_2 norm is used instead of standard ℓ_2 norm for the $\boldsymbol{x}$ update.

In summary, if we use a kernel denoiser for PnP regularization and we make some minor algorithmic adjustments to the PnP updates, the resulting iterates are guaranteed to converge to the minimum of $f + \rho h_{\text{D}}$ as shown in Theorems 2.12 and 2.13. Thus, the seemingly ad-hoc idea of "plugging" a denoiser into the reconstruction process does amount to solving a regularization problem if we use a kernel denoiser. Having been able to associate an optimization problem with the PnP iterations, a natural question is can the optimization problem be solved more efficiently? This would provide an alternate means for performing PnP regularization. We show that this is indeed the case for linear inverse problems.

3.3.2 Reduction to a Linear System

Recall from the previous discussion that PnP regularization amounts to solving the optimization problem

$$\min_{\boldsymbol{x} \in \mathscr{R}(\mathbf{W})} \left\{ \frac{1}{2}||\boldsymbol{y} - \mathbf{A}\boldsymbol{x}||_2^2 + \frac{\rho}{2}\boldsymbol{x}^\top \mathbf{D}(\mathbf{I} - \mathbf{W})\mathbf{W}^\dagger \boldsymbol{x} \right\}. \tag{3.6}$$

It is not obvious if (3.6) has a minimizer since $\mathbf{A}$ has a non-trivial null space [140]. In Chapter 2, we had assumed the existence of a minimizer without proof. We can establish this rigorously.

Proposition 3.5. *The optimization problem* (3.6) *is solvable. In particular, consider the linear system* $\mathbf{F}\boldsymbol{z} = \boldsymbol{b}$*, where*

$$\mathbf{F} = \mathbf{W}^\top \mathbf{A}^\top \mathbf{A}\mathbf{W} + \rho \mathbf{W}^\top \mathbf{D}(\mathbf{I} - \mathbf{W}), \quad \boldsymbol{b} = \mathbf{W}^\top \mathbf{A}^\top \boldsymbol{y}. \tag{3.7}$$

Then $\mathbf{F}\boldsymbol{z} = \boldsymbol{b}$ *is solvable. Moreover,* $\boldsymbol{x}^*$ *is a minimizer of* (3.6) *if and only if* $\boldsymbol{x}^* = \mathbf{W}\boldsymbol{z}$ *where* $\boldsymbol{z}$ *is a solution of* $\mathbf{F}\boldsymbol{z} = \boldsymbol{b}$*.*

We remark that although $\mathbf{W}^\dagger$ appears in the objective function in (3.6), it does not come up in (3.7). This is important since computing $\mathbf{W}^\dagger$ from $\mathbf{W}$ via its eigendecomposition would be impractical given the black-box nature of $\mathbf{W}$ and its size. Note that $\mathbf{W}^\top = \mathbf{D}\mathbf{W}\mathbf{D}^{-1}$. Hence, computing $\mathbf{W}^\top \boldsymbol{x}$ has the same cost as that of computing $\mathbf{W}\boldsymbol{x}$ since $\mathbf{D}$ is a diagonal matrix.

Though the linear system $\mathbf{F}\boldsymbol{z} = \boldsymbol{b}$ is solvable, $\mathbf{F}$ can nevertheless be singular. However, most iterative linear solvers in their original form (e.g. Krylov methods) require $\mathbf{F}$ to be nonsingular for theoretical convergence [171]. The following proposition is useful in this regard.

Proposition 3.6. *Let* $\boldsymbol{e} \in \mathbb{R}^n$ *be the all-ones vector. If* $\mathbf{W}$ *is nonsingular and* $\mathbf{A}\boldsymbol{e} \neq 0$*, then* $\mathbf{F}$ *is nonsingular.*

For deblurring, superresolution and inpainting, $\mathbf{A}$ trivially satisfies the hypothesis in Proposition 3.6 since $\boldsymbol{e}$ does not belong to the nullspace of $\mathbf{A}$. On the other hand, for bilateral and NLM filters, we showed $\mathbf{W}$ to be nonsingular in Theorem 2.16, if the spatial component of the kernel function ϕ is a hat function .

Moreover if $\mathbf{W}$ is nonsingular, $\mathbf{F}\boldsymbol{z} = \boldsymbol{b}$ is equivalent to $\mathbf{C}\boldsymbol{z} = \boldsymbol{d}$, where

$$\mathbf{C} = \mathbf{A}^\top \mathbf{A}\mathbf{W} + \rho \mathbf{D}(\mathbf{I} - \mathbf{W}), \quad \boldsymbol{d} = \mathbf{A}^\top \boldsymbol{y}. \tag{3.8}$$

The advantage with (3.8) compared to (3.7) is that $\mathbf{W}^\top$ does not appear in $\mathbf{C}$. However, unlike $\mathbf{F}$, $\mathbf{C}$ is no longer guaranteed to be symmetric.

3.3.3 Proposed Algorithm

The overall algorithm stemming from the previous discussion is summarized in Algorithm 1.

In the first step of Algorithm 1, the guide image $\hat{\boldsymbol{x}}$ is derived from the measurements $\boldsymbol{y}$ using few (about 5-10) PnP iterations, as done in [79, 172]. This is used to construct the kernel denoiser $\mathbf{W}$ and in turn the operator $\mathbf{C}$. We choose NLM as the preferred kernel denoiser because of its superior regularization capabilities. The final and the core step is the

Algorithm 1: Regularization using a linear solver

Input: Measurements $\boldsymbol{y}$ and forward operator $\mathbf{A}$.
Output: Solution of (3.6).
1 From the guide image $\hat{\boldsymbol{x}}$ from measurements $\boldsymbol{y}$;
2 Construct the denoising operator $\mathbf{W}$ from $\hat{\boldsymbol{x}}$;
3 Use $\mathbf{W}$ and $\mathbf{A}$ to construct the operator $\mathbf{C}$;
4 Solve $\mathbf{C}\boldsymbol{z} = \mathbf{A}^\top \boldsymbol{y}$;
5 Return $\mathbf{W}\boldsymbol{z}$.

solution of $\mathbf{C}\boldsymbol{z} = \mathbf{A}^\top \boldsymbol{y}$. We have used iterative solvers such as LGMRES or GCROT (using the *linalg* module in the *scipy* package), which seem to provide the best trade-off between iteration complexity and the number of iterations required to converge. This is highlighted in Fig. 3.2. We wish to clarify that $\mathbf{W}$, $\mathbf{A}$, and $\mathbf{A}^\top$ in (3.8) are not stored as matrices. Rather, they can be implemented efficiently as input-output black boxes. Indeed, $\mathbf{W}\boldsymbol{x}$ can be computed directly from (2.16). We give a few examples of how to implement $\mathbf{A}$ and $\mathbf{A}^\top$ for different applications. For deblurring, $\mathbf{A}$ and $\mathbf{A}^\top$ can be implemented using spatial convolution (narrow blur) or FFT (wide blur). For superresolution, $\mathbf{A}$ is implemented using convolution with blur kernel followed by decimation operation and $\mathbf{A}^\top$ is implemented using upsampling operation followed by convolution. For inpainting, $\mathbf{A}$ is implemented by choosing pixels that are not missing in the measurements and $\mathbf{A}^\top = \mathbf{A}$.

In particular, $\mathbf{F}$ or $\mathbf{C}$ cannot be computed or stored. Hence, we are forced to use iterative solvers instead of direct solvers for solving corresponding linear systems. For any iterative solver, we require application of $\mathbf{F}\boldsymbol{x}$ to solve linear system in (3.7) (respectively $\mathbf{C}\boldsymbol{x}$ to solve (3.8)) [160]. When $\boldsymbol{x}$ is an image of size 512×512, the time taken to compute $\mathbf{F}\boldsymbol{x}$ (1.8 seconds) is twice the time taken to compute $\mathbf{C}\boldsymbol{x}$ (0.9 seconds). This is because $\mathbf{F}$ has an additional term of $\mathbf{W}^\top$ and hence $\mathbf{F}\boldsymbol{x}$ consists of two denoising operations as compared to one denoising operation for $\mathbf{C}\boldsymbol{x}$. In particular, computational time per iteration for (3.7) is twice that of (3.8).

3.3.4 Comparison with Existing PnP Algorithms

We remark that the optimization in (3.6) can be solved using existing PnP algorithms in Chapter 2 or the proposed linear solver (3.8). Moreover, both are iterative. An obvious question is what advantage do we derive from the latter? In this regard, we wish to discuss the following points.

- For solving linear systems, Krylov solvers like GMRES, GCROT, and LGMRES are known to be superlinearly convergent [171] On the other hand, iterative PnP algorithms such as PnP-ISTA, PnP-FISTA, and PnP-ADMM are only sublinearly convergent [40, 167]. Hence, in theory, Algorithm 1 with Krylov solvers should require fewer iterations

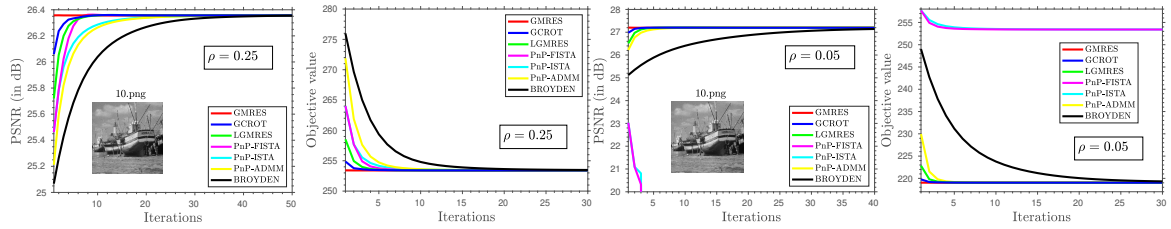


Figure 3.2: Comparison of the efficiency of iterative solvers for image deblurring. A Gaussian blur of 25×25 and standard deviation 1.6 is used and the noise level is $\sigma = 10/255$. The plots show the evolution of PSNR and the objective function (3.6) for $\rho = 0.25$ (left column) and $\rho = 0.05$ (right column). We have used the NLM denoiser for $\mathbf{W}$. The comparison is between PnP-ISTA, PnP-FISTA, PnP-ADMM, and linear solvers GMRES, GCROT, and LGMRES. As expected, the linear solvers exhibit faster convergence than the iterative minimization of (3.6) using PnP algorithms. For example, GCROT takes only 40 percent of iterations than that taken by PnP-ADMM to stabilize. Note that PnP-ISTA and PnP-FISTA diverges at $\rho = 0.05$.

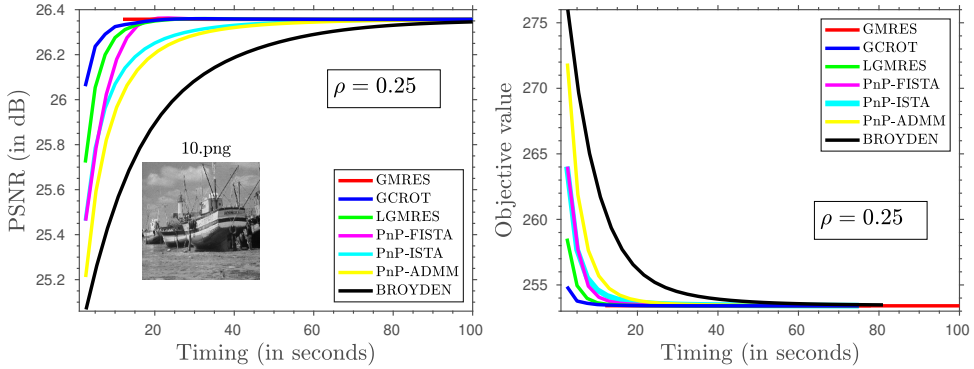


Figure 3.3: Efficiency comparison of the iterative algorithms for the experiment in Fig. 3.2. The linear solvers are faster to stabilize than the iterative minimization of (3.6) using PnP algorithms.

as compared to the PnP algorithms. We have experimented using Krylov solvers GMRES [162], LGMRES [163], GCROT [164] and the quasi-Newton Broyden solver [165, 166]. The evolution of the objective function (and PSNR) for a deblurring experiment is shown in Fig. 3.2 for $\rho \in \{0.05, 0.25\}$. Compared to PnP-ISTA, PnP-FISTA, and PnP-ADMM algorithms, Algorithm 1 indeed converges much faster to a minimizer of $f + \rho h_{\mathrm{D}}$, regardless of the linear solver used. Based on the convergence rate, we can order them as follows: GMRES (12s) > GCROT (2.5s) > LGMRES (2.5s) > PnP-ISTA, PnP-FISTA, PnP-ADMM (2.5s) > Broyden (2.7s), where the per-iteration cost is mentioned within brackets. We note that the per-iteration cost of GCROT and LGMRES is comparable to that of the PnP algorithms, albeit with a superior convergence rate.

- For our proposal, we are required to solve just one linear system. On the contrary, in every iteration of PnP-ADMM (3.5), we need to solve the following linear system as part of the $\boldsymbol{x}$ update (outer iterations):

$$(\mathbf{A}^\top \mathbf{A} + \mathbf{D})\boldsymbol{v}_{k+1} = \mathbf{A}^\top \boldsymbol{y} + \mathbf{D}(\boldsymbol{x}_k - \boldsymbol{z}_k).$$

This in turn requires an iterative solver (inner iterations) for deblurring and superresolution.

- In Fig. 3.2, notice that PnP-ISTA and PnP-FISTA diverges when $\rho = 0.05$. This is because $\rho = 0.05$ violates the bound $\rho \geqslant \sigma_{\max}(\mathbf{D}^{-1/2}\mathbf{A}^\top\mathbf{D}\mathbf{A}\mathbf{D}^{-1/2})$ which is required for convergence as shown in Chapter 2. On the other hand, proposed linear solver works for any $\rho > 0$. This is important since we often obtain better reconstructions for smaller ρ, e.g., PSNR of 27.2 dB at $\rho = 0.05$ compared to 26.35 dB at $\rho = 0.25$.

Table 3.1: Comparison of the PnP algorithms in Chapter 2 and the proposed solver.

Property	PnP algorithms	Proposed	Comments
convergence rate	sublinear convergence	superlinear convergence	superior convergence rate of our method is empirically seen in Fig. 3.2.
cost per iteration	more for PnP-ADMM in most applications since we need to solve a linear system	same as PnP-ISTA	total time taken to stabilize is less for our method as shown in Fig. 3.3.
flexibility of step-size ρ	$\rho > \sigma_{\max}(\mathbf{F})$ is required to guarantee convergence for PnP-ISTA	any $\rho > 0$ can be used	as seen in Fig. 3.2, ρ needs to be small to obtain state-of-the-art results.
existence of minimizer of (3.6)	assumed	proved in proposition 3.5	we can provide certify this for general linear inverse problems.solvability.
unique minimizer of (3.6)	not discussed	proved in proposition 3.6	uniqueness can be certified for deblurring, inpainting and superresolution.

In summary, Algorithm 1 has several computational advantages over the PnP algorithms in Chapter 2, though they solve the same optimization problem; a detailed comparison is provided in Table 3.1.

3.3.5 Comparison of Convergent Linear Solvers

We consider four Krylov techniques for solving (3.8), GMRES, LGMRES, GCROT, and quasi-Newton Broyden, since they come with convergence guarantees and can handle asymmetric systems. We have already shown in Section 3.3.4 that Algorithm 1 has a better convergence rate than PnP algorithms. Further, the evolution of the objective function (and PSNR) for a deblurring experiment is shown in Figures 3.2 and 3.3. Based on convergence rate (number of iterations), we can order them as follows: GMRES > GCROT > LGMRES > Broyden, whereas based on the time taken to stabilize, the ordering is: GCROT > LGMRES > GMRES > Broyden. We note that all four solvers in effect solve the same optimization problem and hence stabilize to the same PSNR (objective) value. We will find a similar order in every application since the convergence rate is intrinsic to the algorithm and does not depend on the

application. It is clear from the empirical analysis that GCROT offers an optimal trade-off between convergence rate and cost per iteration.

Table 3.2: Comparison of PSNR, where we use different number of PnP iterations to obtain the guide image from the measurements.

	Deblurring						Inpainting					
# PnP iterations	0	1	2	3	4	5	0	1	2	3	4	5
barbara	**23.88**	23.81	23.88	23.82	23.81	23.79	24.16	23.10	23.58	24.81	26.11	**27.19**
boat	27.22	27.36	**27.46**	27.30	27.22	27.11	26.93	27.47	27.57	27.70	27.87	**27.99**
cameraman	24.59	24.67	**24.78**	24.68	24.62	24.54	23.55	24.29	24.48	24.64	24.80	**24.93**
couple	26.74	26.92	**27.00**	26.95	26.89	26.82	26.81	27.09	27.21	27.45	27.72	**27.90**
fingerprint	**25.88**	25.78	25.27	24.73	24.36	24.00	23.52	25.14	25.36	25.63	25.94	**26.20**
hill	28.16	28.15	**28.27**	28.15	28.15	28.13	28.86	28.90	28.96	29.06	29.18	**29.27**
house	**29.96**	29.48	29.62	29.37	29.92	29.14	29.34	30.48	30.86	31.32	31.82	**32.17**
lena	30.32	30.24	**30.43**	30.25	30.19	30.08	30.35	31.05	31.23	31.46	31.67	**31.84**
man	28.06	28.06	**28.17**	28.05	28.00	27.91	28.16	28.39	28.48	28.62	28.77	**28.87**

3.3.6 Choice of Guide Image

In Section 3.3.3, the guide image $\boldsymbol{x}$ is derived from the measurements $\boldsymbol{y}$ using 5 to 10 PnP iterations, as done in [79, 172]. Note that the final reconstruction depends on the regularizer $\phi_{\mathbf{W}}$ in (3.3) and hence kernel filter $\mathbf{W}$, and $\mathbf{W}$, in turn, depends on the choice of guide. Intuitively, we should expect better reconstructions if the guide image resembles the ground-truth, i.e., if we use more PnP iterations to compute the guide from the measurements—this is indeed the case with the inpainting results in Table 3.2. However, this is not the case in general. We empirically found that the dependence of the number of PnP iterations (used to compute the guide) on the reconstruction is complicated and depends on the image and application at hand. For example, in line with the deblurring results in Table 3.2, we see that one or two iterations seem to give better results for all deblurring experiments. This is because as iterations to generate the guide image increase, the guide image gets smoothed out and hence final reconstruction performance decreases. On the other hand, for inpainting and superresolution, the reconstruction is seen to improve with the number of PnP iterations. We have demonstrated the same for inpainting in Table 3.2. We wish to investigate this aspect more thoroughly in future work. Furthermore, since the generation of the guide image consumes most of the time in Algorithm 1, as shown in Section 1.4, we need to devise novel methods to generate guide image to optimize the time requirement for Algorithm 1.

3.4 Experimental Results

To understand the reconstruction capability of the proposed algorithm in relation to state-of-the-art methods, we apply our algorithm to three inverse problems—inpainting, superresolution, and deblurring. In step (4) of Algorithm 1, we use GCROT solver (only 5 iterations and hence takes only 10 to 12 seconds for a 512×512 image). All experiments were performed

on a 2.3 GHz, 36 core machine (no GPUs used). The input images in Fig. 3.4 are used for comparisons. Timings are reported to highlight the speedup obtained using our algorithm.

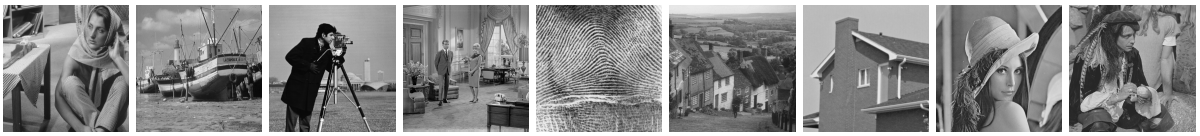
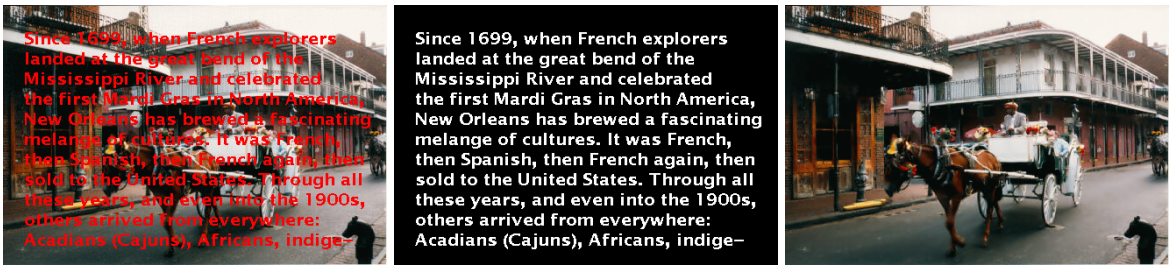
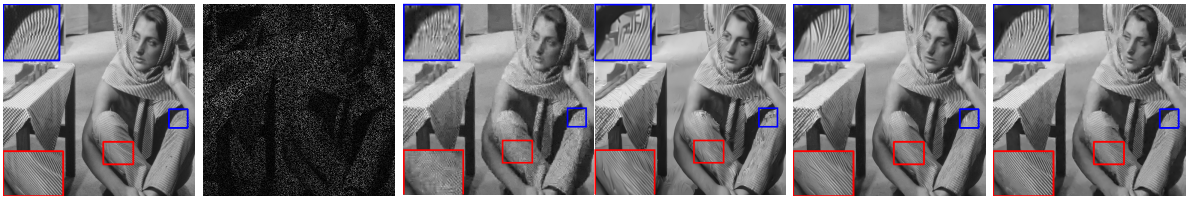


Figure 3.4: Grayscale input images used for quantitative comparisons.



(a) Observation. (b) Mask (c) Output.

Figure 3.5: Text removal using the proposed algorithm. Our method requires solving just one linear system in (3.8) (time taken is 12 seconds). Zoom in to notice the complete removal of the text.



(a) Ground truth (b) Observed (c) EPLL [31] (d) IDBP-CNN [87] (e) PnP-CNN [78] (f) Proposed.

Figure 3.6: Image inpainting from just 20% pixels at $\sigma = 10/255$ noise level. The proposed method ($\rho = 0.05$) is compared with state-of-the-art methods (c)–(e). PSNR and SSIM values: (c) 24.87 dB, 0.729; (d) 26.12 dB, 0.806; (e) 26.99 dB, 0.812; and (f) **27.67** dB, **0.840**. A comparison of the zoomed regions shows the superior reconstruction capability of our method in textured regions.

3.4.1 Inpainting

Refer to Section 2.2.4 for the forward model for inpainting. A sample real-world application for removing text from an image is shown in Fig. 3.5. Further, in Tables 3.3 and 3.4, we present an extensive comparison with state-of-the-art methods for image inpainting using just 20% pixels. Note that we are competitive with CNN-based methods [72] and [87], though we take lesser time. We also highlight that solving the linear system in step 4 of our algorithm takes only 10 to 12 seconds and the remaining time is taken by step 1 to construct the guide image. For the result shown in Fig. 3.6, we see that our method can retrieve thin edges in the image which the compared methods cannot. In Table 3.5, we have shown our algorithm to be

competitive with the best-performing methods on the BSDS300 dataset [139] comprising of over 300 images.

Table 3.3: Experiment 1: Comparison of timing, PSNR, and SSIM for Inpainting with 80 percent missing pixels (Noiseless case).

Method	time (s)	*barbara*	*boat*	*cameraman*	*couple*	*fingerprint*	*hill*	*house*	*lena*	*man*
EPLL [31]	258.1	25.43 0.792	27.51 0.788	24.31 0.816	27.69 0.801	24.20 0.845	29.06 0.776	30.71 0.867	30, 42 0.867	27.94 0.795
IRCNN[72]	31.06	27.34 0.858	27.88 0.809	25.27 0.838	28.23 0.834	25.97 0.887	29.24 0.804	32.21 0.888	31.56 0.889	28.60 **0.830**
IDBP-BM3D [87]	45.5	25.55 0.841	28.51 **0.824**	24.86 0.840	**28.80** 0.846	25.09 0.878	29.74 0.810	**33.78** **0.893**	32.13 0.893	28.65 0.824
IDBP-CNN [87]	39.77	24.29 0.796	27.72 0.803	24.24 0.826	27.98 0.818	25.59 0.865	29.01 0.790	32.14 0.881	31.22 0.880	28.58 **0.830**
Proposed - yaroslavsky [65]	**26.75**	29.14 0.884	27.88 0.767	25.10 0.816	28.07 0.792	26.68 0.913	28.96 0.758	32.12 0.850	31.25 0.853	28.32 0.719
Proposed - bilateral [67]	**26.75**	29.37 0.896	28.24 0.817	24.85 0.842	28.56 0.840	26.34 0.902	29.53 0.823	32.83 0.829	31.98 0.894	28.80 0.819
Proposed - NLM (Laplacian)	**28.15**	29.32 0.890	28.28 0.821	25.07 0.828	28.64 0.845	26.29 0.900	29.75 0.823	33.43 0.895	32.06 0.894	28.82 0.829
Proposed - NLM (Gaussian)	**28.15**	**29.52** **0.900**	**28.56** **0.824**	**25.71** **0.848**	28.72 **0.849**	**26.45** **0.908**	**29.78** **0.825**	33.08 0.891	**32.42** **0.896**	**28.90** 0.829

Table 3.4: Experiment 2: Comparison of timing, PSNR, and SSIM for Inpainting with 80 percent missing pixels (noise level 0.04).

Method	time(s)	*barbara*	*boat*	*cameraman*	*couple*	*fingerprint*	*hill*	*house*	*lena*	*man*
EPLL [31]	257.3	24.76 0.720	26.59 0.720	23.88 0.727	26.70 0.733	23.31 0.812	27.86 0.710	29.10 0.773	28.95 0.777	26.99 0.724
IRCNN [72]	31.10	25.94 0.781	26.86 0.741	**24.75** 0.794	26.98 0.757	24.71 0.834	27.90 0.726	30.61 0.845	29.94 0.833	27.01 0.725
IDBP-BM3D [87]	39.97	25.03 0.755	27.02 0.731	24.68 0.786	**27.22** 0.759	24.99 0.836	28.00 0.708	**31.62** 0.850	30.14 0.835	27.24 0.727
IDBP-CNN [87]	63.10	26.12 0.753	26.95 **0.756**	23.94 0.791	27.04 0.773	24.50 0.857	27.93 **0.734**	31.16 **0.851**	**30.17** **0.849**	27.21 **0.735**
Proposed	**28.30**	**27.67** **0.791**	**27.16** 0.746	24.64 **0.795**	27.08 **0.761**	**25.92** **0.886**	**28.04** 0.732	30.70 0.835	30.10 0.833	**27.39** 0.717

At this point, we wish to justify our choice of NLM as the preferred kernel denoiser. In Table 3.3, we have compared the results using different kernel denoisers: bilateral, Yaroslavsky, standard NLM (Gaussian kernel), and NLM (Laplacian kernel) and it is apparent that we get the best results for standard NLM. This is not surprising since the standard NLM is known to enjoy superior denoising capabilities to other kernel denoisers [68]. Hence, we

Table 3.5: Comparison of PSNR and SSIM for different experiments for Inpainting averaged over BSDS300 dataset.

Methods	PnP-BM3D [100]	PnP-DnCNN [78]	Proposed
Experiment 1	27.01 0.8232	27.14 **0.8265**	**27.15** 0.8242
Experiment 2	25.59 0.7485	**26.05** **0.7584**	26.00 0.7493

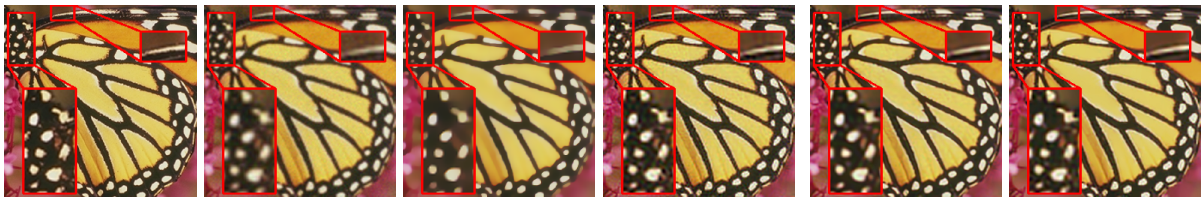
Table 3.6: Comparison of timing and PNSR for superresolution (9×9 Gaussian blur with standard deviation 1, and noise level 5/255).

Methods	time (s)	*barbara*	*boat*	*cameraman*	*couple*	*fingerprint*	*hill*	*house*	*lena*	*man*
$K = 2$										
SR[146]	298.6	23.61	26.25	23.71	26.15	23.80	27.41	27.71	28.07	26.99
GPR[147]	390.1	23.82	26.81	23.91	26.63	24.05	28.38	29.16	29.54	27.78
NCSR[123]	256.2	24.67	28.40	26.22	28.02	27.74	28.50	29.85	30.43	28.75
PnP-BM3D [100]	48.65	24.64	29.41	26.73	29.22	28.82	29.82	32.65	32.76	29.66
PnP-DnCNN [78]	39.45	24.50	**29.57**	**27.36**	29.29	29.06	29.97	32.12	32.58	**29.98**
Proposed	**28.12**	**24.96**	**29.57**	26.74	**29.38**	**29.28**	**30.07**	**32.68**	**32.83**	**29.98**
$K = 4$										
SR[146]	298.6	20.67	21.30	18.86	21.51	16.37	23.15	22.19	22.85	22.26
GPR[147]	390.1	21.55	22.68	19.90	22.77	17.70	24.57	23.51	24.37	23.63
NCSR[123]	256.2	22.86	24.38	22.04	24.18	22.31	25.01	26.30	26.90	25.41
PnP-BM3D [100]	48.65	23.62	**25.75**	23.06	**25.30**	23.48	27.17	**29.14**	29.42	26.86
PnP-DnCNN [78]	39.45	23.66	25.73	**23.39**	25.27	23.61	27.19	29.05	29.44	26.92
Proposed	**28.12**	**23.67**	25.64	23.05	**25.30**	**23.65**	**27.20**	28.46	**29.45**	**26.93**

have used the standard NLM denoiser for all experiments. This behaviour is due to the fine-tuning of smoothing parameters for NLM denoiser, such that we get superior reconstruction capabilities.

3.4.2 Superresolution

Refer to Section 2.2.4 for the forward model for superresolution. In Table 3.6, we compare our method with state-of-the-art methods at $K = 2$ and 4 using a Gaussian blur. Similar to inpainting, we are able to compete with state-of-the-art algorithms while being faster than them. In Fig. 3.7, we compare our method with PnP-ADMM by plugging different denoisers. Notice that similar to PnP regularization using DnCNN denoiser [78], our method gives the best result. In this experiment, we can restore even fine features in the image unlike BM3D, and do not oversmooth the image, unlike TV. We remark that, unlike our proposed algorithm, PnP regularization using DnCNN does not come with any optimality guarantee. Note that since we apply denoiser in every iteration, the final reconstruction image can have the textures smoothed out or the loss of fine features. For superresolution, PnP with DnCNN and the proposed method seem to balance between oversmoothing and restoring fine features.



(a) Ground truth (b) Bicubic [150] (c) TV [150] (d) BM3D [70] (e) DnCNN [78] (f) Proposed

Figure 3.7: Results for 2× image superresolution (using 9×9 Gaussian blur with standard deviation 1 and noise level 5/255). Compared to BM3D (see zoomed regions), the quality is perceptibly better for PnP regularization, both for DnCNN and the proposed method ($\rho = 2$). PSNR(dB), SSIM values: (b) 22.57, 0.78 (c) 24.3, 0.83 (d) 26.2, 0.86 (e) **27.4**, 0.89; and (f) 27.03, **0.90**.

(a) Measurements (b) IRCNN [72] (c) Proposed

Figure 3.8: Image deblurring with an asymmetric blur kernel in [72]. Our result is visibly better than the learning-based method [72] (no sharp artefacts).

3.4.3 Deblurring

Refer to Section 2.2.4 for the forward model for deblurring. We show a deblurring application in Fig. 3.8 where the blur kernel is asymmetric. Our method does not produce ringing artefacts as seen in the state-of-the-art method [72]. Note that NLM has a parameter to control smoothing the image (N_s in (2.20)). We found empirically that choosing N_s appropriately (3 in this case) smoothens the ringing artefacts but preserves the sharp edges. In Table 3.7, we perform three experiments as in [152] for different Gaussian blur and noise standard deviation. The results are averaged over the images in Fig. 3.4.

- (I) Gaussian blur 25×25 and standard deviation 1.6, and noise $\sigma = 0.04$.
- (II) Gaussian blur 25×25 and standard deviation 1.6, and noise $\sigma = 2/255$.
- (III) Gaussian blur 25×25 and standard deviation 3, and noise $\sigma = 0.04$.

It is seen from Table 3.7 that the proposed method is competitive with state-of-the-art deblurring methods [123, 72]. In Table 3.8, we have shown that our method is generally competitive and can sometimes outperform the best-performing methods [123, 72] on the BSDS300 dataset.

3.4.4 Discussion

It might seem surprising that NLM is able to compete with a pretrained deep denoiser for image regularization; after all, the denoising quality of DnCNN is generally a few dBs better than

Table 3.7: Comparison of averaged PSNR for deblurring.

Method	Exp-I	Exp-II	Exp-III
EPLL [31]	24.04	26.64	21.36
MLP [152]	24.76	27.23	22.20
FlexISP [128]	24.32	26.84	21.99
LUT [154]]	24.17	26.60	21.73
[101]	24.51	27.08	21.83
DEB-BM3D [70]	24.19	26.30	21.48
NCSR [123]	26.62	30.03	24.51
IDD-BM3D [155]	24.68	27.13	21.99
IRCNN [72]	**27.93**	**30.43**	**25.67**
Proposed	**27.70**	**30.40**	**25.62**

Table 3.8: Comparison of PSNR and SSIM for different experiments for Deblurring averaged over BSDS300 dataset.

Methods	NCSR [123]	IRCNN [72]	Proposed
Exp-I	25.56 0.6707	**26.75** 0.7300	26.25 **0.7351**
Exp-II	28.97 0.8338	**29.25** 0.8340	28.57 **0.8371**
Exp-III	24.01 0.5527	**24.83** 0.6441	24.33 **0.6473**

NLM. In this regard, we note that the exact relation between denoising capacity and the final restoration quality (within the PnP framework) is not well understood. For example, although DnCNN is more powerful than BM3D [9], it is known that plugging BM3D denoiser within a PnP algorithm can produce better results than DnCNN [78, 173]. Similarly, NLM has been shown to outperform BM3D (which is more powerful than NLM) for some applications [79, 174]. As shown in Chapter 2 and [72], in most of the application settings, CNN denoisers demonstrate better regularization capabilities than kernel denoisers. This is one of the main motivations behind the construction of a provably averaged denoiser in Chapter 4.

3.5 Conclusion

We first showed that PnP regularization using kernel denoisers amounts to solving the classical regularization problem of minimizing $f + g$, where f is the loss term and g is a convex quadratic regularizer (derived from data). We went on to show that for linear inverse problems with quadratic f, minimizing $f + g$ can be reduced to a linear system, which is solvable

and admits a unique solution for deblurring, superresolution, and inpainting. Instead of performing PnP iterations, we proposed to directly solve this linear system using efficient Krylov solvers. The proposed solver converges at a superlinear rate which is a big jump from the sublinear convergence guarantee of first-order PnP algorithms. We validated the speedup in practice using deblurring, superresolution, and inpainting experiments. In terms of reconstruction quality, we were able to get close to deep learning methods, and hence our proposed linear solver could be the algorithm of choice for real-time performance. However, the method proposed in this Chapter has the following limitation. The time taken to generate the guide image is more than solving the linear system. Hence, we need to come up with a fast algorithm to generate the guide image, which in turn will facilitate real-time performance for the proposed method.

3.6 Appendix

In this section, we state and prove the technical results in Section 3.3. We first recall a few results from linear algebra and convex optimization.

3.6.1 Preliminaries

We use $\mathcal{N}(\mathbf{Z})$ and $\mathcal{R}(\mathbf{Z})$ to denote the null space and range spaces of a matrix $\mathbf{Z}$.

Proposition 3.7. *Let $\mathbb{R}^n = U_1 \oplus V_1 = U_2 \oplus V_2$, where $\oplus$ denotes orthogonal direct sum. Then $U_1 \subseteq U_2$ implies $V_2 \subseteq V_1$.*

Proof. Fix $\boldsymbol{x} \in V_2$. By our assumption, $\boldsymbol{x} \perp \boldsymbol{y}$ for all $\boldsymbol{y} \in U_2$. Since $U_1 \subseteq U_2$, this mean that $\boldsymbol{x}$ belongs to the orthogonal complement of U_1, which is V_1 by assumption. □

Proposition 3.8. *Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n\times n}$ be symmetric positive semidefinite matrices. Then $\mathcal{N}(\mathbf{A}+\mathbf{B}) \subseteq \mathcal{N}(\mathbf{A}) \cup \mathcal{N}(\mathbf{B})$ and $\mathcal{R}(\mathbf{A}) \cup \mathcal{R}(\mathbf{B}) \subseteq \mathcal{R}(\mathbf{A}+\mathbf{B})$.*

Proof. Let $\boldsymbol{x} \in \mathcal{N}(\mathbf{A}+\mathbf{B})$. Then $\boldsymbol{x}^\top(\mathbf{A}+\mathbf{B})\boldsymbol{x} = 0$. However, by assumption, $\boldsymbol{x}^\top \mathbf{A}\boldsymbol{x} \geqslant 0$ and $\boldsymbol{x}^\top \mathbf{B}\boldsymbol{x} \geqslant 0$ for all $\boldsymbol{x}$. Thus $\boldsymbol{x}^\top(\mathbf{A}+\mathbf{B})\boldsymbol{x} = 0$ only if $\boldsymbol{x}^\top \mathbf{A}\boldsymbol{x} = 0$ and $\boldsymbol{x}^\top \mathbf{B}\boldsymbol{x} = 0$, which further implies that $\mathbf{A}\boldsymbol{x} = \mathbf{B}\boldsymbol{x} = 0$. Hence, we conclude that $\boldsymbol{x} \in \mathcal{N}(\mathbf{A}) \cap \mathcal{N}(\mathbf{B})$ for all $\boldsymbol{x} \in \mathcal{N}(\mathbf{A}+\mathbf{B})$.

Since $\mathbf{A}, \mathbf{B}$ and $\mathbf{A}+\mathbf{B}$ are symmetric, $\mathbb{R}^n = \mathcal{R}(\mathbf{A}) \oplus \mathcal{N}(\mathbf{A}) = \mathcal{R}(\mathbf{B}) \oplus \mathcal{N}(\mathbf{B}) = \mathcal{R}(\mathbf{A}+\mathbf{B}) \oplus \mathcal{N}(\mathbf{A}+\mathbf{B})$. We just proved that $\mathcal{N}(\mathbf{A}+\mathbf{B}) \subseteq \mathcal{N}(\mathbf{A})$ and $\mathcal{N}(\mathbf{A}+\mathbf{B}) \subseteq \mathcal{N}(\mathbf{B})$. Hence, by Proposition 3.7, $\mathcal{R}(\mathbf{A}) \subseteq \mathcal{R}(\mathbf{A}+\mathbf{B})$ and $\mathcal{R}(\mathbf{B}) \subseteq \mathcal{R}(\mathbf{A}+\mathbf{B})$, so $\mathcal{R}(\mathbf{A}) \cup \mathcal{R}(\mathbf{B}) \subseteq \mathcal{R}(\mathbf{A}+\mathbf{B})$. □

The following is a first-order optimality condition for convex functions [16].

Theorem 3.9. *Let $\psi : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ be convex. Moreover, assume that there exists a differentiable function $q : \mathbb{R}^n \to \mathbb{R}$ such that q restricted to $\mathrm{dom}(\psi) := \{\boldsymbol{x} \in \mathbb{R}^n : \psi(\boldsymbol{x}) < \infty\}$ equals ψ. Then $\boldsymbol{x}^* \in \mathbb{R}^n$ is a minimizer of ψ on $\mathbb{R}^n$ if and only if*

$$\forall \boldsymbol{x} \in \mathrm{dom}(\psi): \quad \nabla q(\boldsymbol{x}^*)^\top(\boldsymbol{x} - \boldsymbol{x}^*) \geqslant 0.$$

3.6.2 Proof of Proposition 3.3

Recall that $h_{\mathrm{D}}(x) = (1/2)x^{\top}\mathbf{D}(\mathbf{I}-\mathbf{W})\mathbf{W}^{\dagger}x$ for all $x \in \mathscr{R}(\mathbf{W})$. Since $\mathscr{R}(\mathbf{W})$ is nonempty, h_{D} is proper.

From the definitions of $\mathbf{W}$ and $\mathbf{W}^{\dagger}$, we can easily check that $\mathbf{D}(\mathbf{I}-\mathbf{W})\mathbf{W}^{\dagger} = \mathbf{P}\Sigma\mathbf{P}^{\top}$, where $\mathbf{P} := \mathbf{D}^{1/2}\mathbf{Q}$ and Σ is diagonal where $\Sigma_{ii} := (1/\Lambda_{ii}) - 1$ if $\Lambda_{ii} > 0$, and $= 0$ otherwise. Clearly, $\mathbf{D}(\mathbf{I}-\mathbf{W})\mathbf{W}^{\dagger}$ is symmetric. Moreover, it can be shown that the eigenvalues of $\mathbf{W}$ are always in $[0,1]$; e.g., see [22, Proposition 3.12]. This means that $\Sigma_{ii} \geqslant 0$ for all i, and hence $\mathbf{D}(\mathbf{I}-\mathbf{W})\mathbf{W}^{\dagger}$ is positive semidefinite, which further implies that h_{D} is nonnegative and convex.

Since h_{D} is continuous on its domain $\mathscr{R}(\mathbf{W})$ and $\mathscr{R}(\mathbf{W})$ is a closed set in $\mathbb{R}^n$, h_{D} is closed [16, Lemma 1.24].

3.6.3 Proof of Theorem 3.4

Let $\psi : \mathbb{R}^n \to \mathbb{R}$ be defined as

$$\psi(z) = \frac{1}{2}||z-x||_{\mathbf{D}}^2 + h_{\mathrm{D}}(z) = \begin{cases} q(z), & \text{if } z \in \mathscr{R}(\mathbf{W}), \\ \infty, & \text{otherwise.} \end{cases}$$

where

$$q(z) := \frac{1}{2}(z-x)^{\top}\mathbf{D}(z-x) + \frac{1}{2}z^{\top}\mathbf{D}(\mathbf{I}-\mathbf{W})\mathbf{W}^{\dagger}z.$$

Following the definition of the scaled proximal operator in equation (7), we need to show that for any $x \in \mathbb{R}^n$, $\mathbf{W}x$ is a minimizer of ψ.

Now, it follows from Proposition 3.3 that h_{D} is convex and $||z-x||_{\mathbf{D}}^2$ is convex since $\mathbf{D}$ is positive definite. Hence, ψ is convex. On the other hand, q is differentiable and

$$\nabla q(z) = \mathbf{D}(z-x) + \mathbf{D}(\mathbf{I}-\mathbf{W})\mathbf{W}^{\dagger}z.$$

Also, its restriction to $\mathrm{dom}(\psi) = \mathscr{R}(\mathbf{W})$ equals ψ. Hence, by Theorem 3.9, the optimality condition becomes

$$\begin{aligned} &\forall u \in \mathbb{R}^n : \ \nabla q(\mathbf{W}x)^{\top}(\mathbf{W}u - \mathbf{W}x) \\ &= (u-x)^{\top}\mathbf{W}^{\top}\mathbf{D}\Big[(\mathbf{W}-\mathbf{I}) + (\mathbf{I}-\mathbf{W})\mathbf{W}^{\dagger}\mathbf{W}\Big]x \geqslant 0. \end{aligned}$$

Now, since $\mathbf{W} = \mathbf{W}\mathbf{W}^{\dagger}\mathbf{W}$ and $\mathbf{W}^{\top}\mathbf{D} = \mathbf{D}\mathbf{W}$, we have

$$\mathbf{W}^{\top}\mathbf{D}\Big[(\mathbf{W}-\mathbf{I}) + (\mathbf{I}-\mathbf{W})\mathbf{W}^{\dagger}\mathbf{W}\Big] = 0.$$

We have thus shown that $\mathbf{W}x$ satisfies the optimality condition and this completes the proof.

3.6.4 Proof of Proposition 3.5

Note that $\boldsymbol{x}^* \in \mathscr{R}(\mathbf{W})$ is a minimizer of the objective in (9) if and only if $\boldsymbol{x}^* = \mathbf{W}\boldsymbol{z}^*$, where $\boldsymbol{z}^*$ is a minimizer of

$$\theta(\boldsymbol{z}) := \frac{1}{2}||\boldsymbol{y} - \mathbf{A}\mathbf{W}\boldsymbol{z}||_2^2 + \frac{\rho}{2}\boldsymbol{z}^\top \mathbf{W}^\top \mathbf{D}(\mathbf{I} - \mathbf{W})\boldsymbol{z}.$$

Since θ is convex and differentiable in $\boldsymbol{z}$, it has a minimizer if and only if there exists $\boldsymbol{z}^* \in \mathbb{R}^n$ such that $\nabla\theta(\boldsymbol{z}^*) = \mathbf{0}$. However, we can write $\nabla\theta(\boldsymbol{z}) = \mathbf{F}\boldsymbol{z} - \boldsymbol{b}$, where $\boldsymbol{b} = \mathbf{W}^\top \mathbf{A}^\top \boldsymbol{y}$ and $\mathbf{F}$ is of the form $\mathbf{F} = \mathbf{Z} + \mathbf{Y}$ where

$$\mathbf{Z} = \mathbf{W}^\top \mathbf{A}^\top \mathbf{A}\mathbf{W} \quad \text{and} \quad \mathbf{Y} = \rho\mathbf{W}^\top \mathbf{D}(\mathbf{I} - \mathbf{W}).$$

Thus, θ has a minimizer if and only if the equation $\mathbf{F}\boldsymbol{z} = \boldsymbol{b}$ is solvable, i.e., $\boldsymbol{b} \in \mathscr{R}(\mathbf{A})$. We next show that this is indeed the case.

By definition, $\boldsymbol{b}$ is in the range of $\mathbf{W}^\top \mathbf{A}^\top$. However, note that $\mathscr{R}(\mathbf{W}^\top \mathbf{A}^\top) = \mathscr{R}(\mathbf{Z})$, so it suffices to show that that $\mathscr{R}(\mathbf{Z}) \subseteq \mathscr{R}(\mathbf{F})$. But this follows from Proposition 3.8 since $\mathbf{Z}$ and $\mathbf{Y}$ are symmetric positive semidefinite and $\mathbf{F} = \mathbf{Y} + \mathbf{Z}$. This completes the proof.

3.6.5 Proof of Proposition 3.6

Let $\mathbf{Y}$ and $\mathbf{Z}$ be as in the proof of Proposition 3.5 and $\mathbf{F} = \mathbf{Y} + \mathbf{Z}$. Since $\mathbf{Z}$ and $\mathbf{Y}$ are symmetric positive semidefinite, by Proposition 3.8, we can conclude that $\mathbf{A}$ is invertible if $\mathrm{N}(\mathbf{Z}) \cap \mathrm{N}(\mathbf{Y}) = \{\mathbf{0}\}$. Now, it can be shown that $\mathbf{W}$ is irreducible, nonnegative, and row stochastic [36, Chapter 8]. Hence, by the Perron-Frobenius theorem, $\boldsymbol{e}$ is the Perron vector and $\mathbf{W}\boldsymbol{e} = \boldsymbol{e}$. Also, $\mathbf{W}$ is nonsingular by hypothesis. Hence $\mathrm{N}(\mathbf{I} - \mathbf{W})$ consists of scalar multiples of the Perron vector $\boldsymbol{e}$. Since $\mathbf{W}$ is invertible by hypothesis and $\mathbf{D}$ is invertible by construction, we have $\mathrm{N}(\mathbf{Y}) = \mathrm{N}(\rho\mathbf{W}^\top \mathbf{D}(\mathrm{I} - \mathbf{W})) = \{t\boldsymbol{e} : t \in \mathbb{R}\}$. Since $\mathbf{W}\boldsymbol{e} = \boldsymbol{e}$ and $\mathbf{A}\boldsymbol{e} \neq \mathbf{0}$ by hypothesis, $\mathbf{A}\mathbf{W}\boldsymbol{e} = \mathbf{A}\boldsymbol{e} \neq \mathbf{0}$ and $\mathrm{N}(\mathbf{A}\mathbf{W}) = \mathrm{N}(\mathbf{W}^\top \mathbf{A}^\top \mathbf{A}\mathbf{W}) = \mathrm{N}(\mathbf{Z})$. Hence $\mathrm{N}(\mathbf{Z}) \cap \mathrm{N}(\mathbf{Y}) = \{\mathbf{0}\}$.

Construction of Averaged Deep Denoisers for Image Regularization

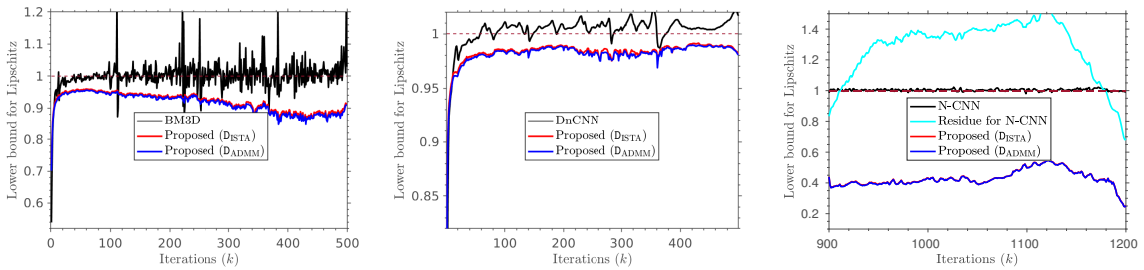
Abstract. *Plug-and-Play (PnP) and Regularization by Denoising (RED) leverage the power of modern denoisers for image regularization, and deliver state-of-the-art reconstructions using CNN denoisers. Recently, it was observed in many works, including our results in Chapter 2, that iterate convergence of PnP and RED can be guaranteed if the denoiser is averaged or nonexpansive. However, integrating nonexpansivity with gradient-based learning is a challenging task—checking nonexpansivity is known to be computationally intractable. Using numerical examples, we show that existing CNN denoisers violate the nonexpansive property and can cause the PnP iterations to diverge. Algorithms for training nonexpansive denoisers either cannot guarantee nonexpansivity of the final denoiser or are computationally intensive. In this work, we propose to construct averaged (contractive) image denoisers by unfolding ISTA and ADMM iterations applied to wavelet denoising and demonstrate that their regularization capacity for PnP and RED can be matched with CNN denoisers. To the best of our knowledge, this is the first work to propose a simple framework for training provably averaged (contractive) denoisers using unfolding networks.*

4.1 Introduction

When the data-fidelity term is convex, it is known that nonexpansive [78] or averaged denoisers (our results in Chapter 2) effect convergence of PnP algorithms. In particular, averaged denoisers are sufficient for convergence of PnP-ISTA (refer Theorem 2.8) and nonexpansive denoisers are sufficient for convergence of RED algorithms [92, 97, 115], PnP-ISTA for compressive sensing [106], variant of PnP-ISTA in [92]. As discussed in Chapter 1, there has been significant research on training averaged or more generally nonexpansive CNN denoisers.

Key Notation difference from other Chapters: *We replace ρ^{-1} in Chapters 1, 2 and 3 with γ in this chapter. This is mandatory to understand the architecture of the trained denoiser. In this chapter, $||\cdot||$ is used to represent euclidean norm. In particular, the subscript 2 is dropped from the notation $||\cdot||_2$ (used in Chapters 2 and 3) to avoid notation clutter.*

The work in this chapter is motivated by the observation that existing CNN denoisers violate the nonexpansive property (see Fig. 4.1). In fact, existing methods for training nonexpansive denoisers either do not guarantee the final network to be nonexpansive or are computationally expensive. For example, power iterations are used for approximate computation of the largest singular value in [78, 119]. Hence, the network is not guaranteed to be nonexpansive at each epoch. On the other hand, the methods in [99, 117, 118] for constraining the convolutional layers are computationally expensive— for example, the time taken for [117] is about nine-fold that of training without constraints. In particular, the projection in [117] requires the full singular value decomposition (SVD) of the convolutional layers after every gradient step, and this requires the FFT of the filters followed by the SVD of a large matrix derived from the FFTs. Similarly, the projection step in [118] requires the FFT of the filters in every convolutional layer. The training mechanism in [99] requires us to solve an optimization problem to impose orthogonality constraints on the convolutional layers.



(a) BM3D. (b) DnCNN. (c) N-CNN [78]

Figure 4.1: Violation of nonexpansive property: For superresolution, PnP-ISTA is run for 500 iterations by plugging different denoisers, and the lower bound for Lipschitz is plotted against iterations. Lipschitz is defined as supremum of $||\mathsf{D}(\boldsymbol{u})-\mathsf{D}(\boldsymbol{v})||/||\boldsymbol{u}-\boldsymbol{v}||$, for all possible images $\boldsymbol{u}$ and $\boldsymbol{v}$. Thus by considering successive iterates $\boldsymbol{x}_k$ as $\boldsymbol{u}$ and $\boldsymbol{x}_{k+1}$ as $\boldsymbol{v}$, various estimates of lower bound for Lipschitz are obtained. Note that irrespective of the denoiser used, the lower bound for Lipschitz is greater than 1 for many iterations, thereby proving that actual Lipschitz is greater than 1 and hence denoisers are not nonexpansive. Whereas, for the same set of inputs, the lower bound for Lipschitz in the case of proposed denoisers is always strictly less than 1, irrespective of the plugged denoiser.

We also highlight a technical issue that has been sidelined in existing works [78, 117, 99, 118, 119], where filters in CNN denoisers are trained on images of fixed size but deployed on images of arbitrary size. Since the filters are zero-padded to match the image size, their Lipschitz constant depends on the image size. For example, consider the 3×3 Sobel filter $[1, 0, -1; 2, 0, -2; 1, 0, -1]$. The Lipschitz constant of this filter applied to images of size 25×25 is 7.9842, whereas the Lipschitz constant of the same filter for 256×256 images is 8. Thus, a guarantee that the Lipschitz bound is independent of the image size is not available for existing CNN denoisers.

To substantiate our point, we present numerical evidence (see Fig. 4.1) showing that BM3D [70], DnCNN [9], and the denoiser in [78] referred to as N-CNN, violate the nonexpansive

property. That is, we produce images $\boldsymbol{u}$ and $\boldsymbol{v}$ such that $||\mathrm{D}(\boldsymbol{u})-\mathrm{D}(\boldsymbol{v})||/||\boldsymbol{u}-\boldsymbol{v}||$ is greater than 1. The detailed procedure used to generate these counterexamples is described in Section 4.5.1. We note that such counterexamples have also been reported in [92]. Note that the denoiser N-CNN is trained for its residual to be nonexpansive. In Fig. 4.1(c), both N-CNN and its residual are shown to violate the nonexpansive property.

Table 4.1: Values of $||\boldsymbol{x}_k-\boldsymbol{x}_{k-1}||$ and PSNR (dB) for PnP-ISTA with DnCNN and proposed denoisers across iterations

Denoiser		Number of iterations				
		10	25	50	75	100
DnCNN	$\|\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|\|_2$	146.00	146.58	234.55	2300	2207248
	PSNR	16.54	14.79	−4.09	−45.14	−84.68
Proposed ($\mathrm{D}_{\mathrm{ISTA}}$)	$\|\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|\|_2$	74.06	30.65	7.67	1.99	0.52
	PSNR	20.42	22.96	23.69	23.74	23.74
Proposed ($\mathrm{D}_{\mathrm{ADMM}}$)	$\|\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|\|_2$	74.09	30.67	7.68	1.99	0.55
	PSNR	20.43	23.00	23.74	23.79	23.79

Table 4.2: $||\boldsymbol{x}_k-\boldsymbol{x}_{k-1}||$ for PnP-ISTA with different denoisers.

Denoiser	iterations (k)					
	10	50	100	500	1000	3000
N-CNN [78]	0.54	0.20	0.19	0.36	23.45	$3e{+}12$
Proposed ($\mathrm{D}_{\mathrm{ISTA}}$)	0.31	0.15	0.11	0.0602	0.0513	$2e$-10
Proposed ($\mathrm{D}_{\mathrm{ADMM}}$)	0.32	0.16	0.11	0.0658	0.0562	$2e$-10

We note that nonexpansivity is just a sufficient condition for convergence. However, as shown in Tables 4.1 and 4.2, PnP algorithms may in fact fail to converge if the plugged denoiser is not nonexpansive; the algorithm returns poor reconstructions in such cases (see Table 4.1). The procedure used to generate the results in Tables 4.1 and 4.2 can be found in Section 4.5.1. It has been empirically observed that even RED algorithms can diverge if denoiser is not nonexpansive. (refer Table II in [97]).

We summarize our observations below:

- The Lipschitz bound of a CNN (trained with filters of fixed size) depends on the size of the input image and is difficult to predict.
- Existing CNN denoisers violate the nonexpansive property. Loss of nonexpansivity of the denoiser can result in divergence of the PnP iterates and poor PSNR values.
- Existing algorithms to train nonexpansive CNN denoisers either lack the guarantee that the final denoiser is nonexpansive or are computationally expensive. In fact, coming up with efficient training algorithms to learn provably nonexpansive denoisers is a difficult task.

4.2 Contractive & Averaged Operators

In this section, we look at contractive and averaged operators and highlight some relevant properties of such operators. These are used to motivate the construction of the proposed denoiser. Importantly, we point out the connection between such operators and fixed-point convergence of PnP algorithms.

We first recall the definition of averaged operators as in Definition 2.1.

Definition 4.1. *An operator* $\mathrm{T}:\mathbb{R}^n\to\mathbb{R}^n$ *is said to be* β*-Lipschitz for some* $\beta>0$ *if*

$$\forall\, \boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^n:\quad \|\mathrm{T}(\boldsymbol{x})-\mathrm{T}(\boldsymbol{y})\|\leqslant\beta\,\|\boldsymbol{x}-\boldsymbol{y}\|\,.$$

In particular, T *is said to be nonexpansive if* $\beta=1$*, and* T *is contractive if* $\beta\in[0,1)$*. An operator* T *is said to be* θ*-averaged, with relaxation parameter* $\theta\in[0,1)$*, if we can write* $\mathrm{T}=(1-\theta)\mathrm{I}+\theta\mathrm{N}$*, where* N *is nonexpansive.*

It is evident from the above definitions that an averaged operator is nonexpansive. The converse is however not true; e.g., the operator $-\mathrm{I}$ is nonexpansive but not averaged [114]. On the other hand, every contractive operator is necessarily averaged, but not every averaged operator is contractive. Thus, contractive operators form a strict subset of averaged operators, which itself is a strict subset of nonexpansive operators [114]. The gradient and proximal operators $\mathrm{I}-\gamma\nabla f$ and $\mathrm{Prox}_{\gamma f}$ associated with a convex function f are averaged. The precise results are collected below.

Lemma 4.2. *(a) Let* $f:\mathbb{R}^n\to\mathbb{R}$ *be convex and* ∇f *be* β*-Lipschitz. Then the operator* $\mathrm{I}-\gamma\nabla f$ *is* $(\beta\gamma/2)$*-averaged for* $0\leqslant\gamma<2/\beta$*.*

(b) Moreover, if f *is* μ*-strongly convex, then* $\mathrm{I}-\gamma\nabla f$ *is contractive for* $0<\gamma<2/\max(\beta,\mu)$*.*

Lemma 4.3. *Let* $f:\mathbb{R}^n\to\mathbb{R}$ *be convex and* $\gamma>0$*. Then the operator* $\mathrm{Prox}_{\gamma f}$ *is* $1/2$*-averaged.*

We recall that Lemma 4.2(a) is indeed Lemma 2.19 in Chapter 2 which is proved in the Appendix of Chapter 2. Proof of Lemma 4.2(b) is given in this chapter's Appendix. See [114, Proposition 2.28, Proposition 4.4] for Lemma 4.3. We will later use Lemmas 4.2 and 4.3 to show that a single layer of our denoising network is either contractive or averaged.

Averaged operators are closed under composition. The exact relaxation parameter of the composition is described in Lemma 4.4. Contractive operators are also closed under composition. We will use these observations to show that if the layers of the denoiser are averaged (contractive), then the end-to-end denoiser is averaged (contractive).

Lemma 4.4. *Let* $\mathrm{T}_1,\ldots,\mathrm{T}_m$ *be operators on* $\mathbb{R}^n$ *that are* θ*-averaged. Then their composition* $\mathrm{T}=\mathrm{T}_1\circ\mathrm{T}_2\circ\cdots\circ\mathrm{T}_m$ *is* $m\theta/\big(1+(m-1)\theta\big)$*-averaged. If each* T_i *is contractive, then their composition* T *is contractive.*

See [114, Proposition 4.46] for a proof. We next look at the average of such operators.

Lemma 4.5. *Let* $\mathrm{T}_1, \mathrm{T}_2, \dots \mathrm{T}_k$ *be* θ*-averaged (contractive) operators. Then* $(1/k)(\mathrm{T}_1 + \mathrm{T}_2 \cdots + \mathrm{T}_k)$ *is* θ*-averaged (contractive).*

Proof of this result can be found in [114, Proposition 4.42]. Later, we will use this to construct an averaged (contractive) image-level denoiser from an averaged (contractive) patch-based denoiser.

4.3 Fixed Point Convergence

Contractive and averaged operators can be used to establish fixed-point convergence of classical proximal (e.g. ISTA and ADMM) and PnP algorithms (as shown in Theorems 2.8 and 2.9). This is done by expressing the updates in ISTA and ADMM as a mapping $\boldsymbol{x} \mapsto \mathrm{T}(\boldsymbol{x})$, where T is contractive or averaged depending on the loss f and the regularizer g. For example, the operator in ISTA in (1.10) is

$$\mathrm{T}_{\mathrm{ISTA}} = \mathrm{Prox}_{\gamma g} \circ (\mathrm{I} - \gamma \nabla f), \tag{4.1}$$

and the operator in ADMM in (1.12) is

$$\mathrm{T}_{\mathrm{ADMM}} = \frac{1}{2}\mathrm{I} + \frac{1}{2}(\mathrm{R}_{\gamma g} \circ \mathrm{R}_{\gamma f}), \tag{4.2}$$

where

$$\mathrm{R}_f = 2\mathrm{Prox}_f - \mathrm{I}$$

is reflected proximal operator of f. More precisely, the ISTA update in (1.10) can be expressed as $\boldsymbol{x}_{k+1} = \mathrm{T}_{\mathrm{ISTA}}(\boldsymbol{x}_k)$. For the ADMM update, we have the following result in Lemma 2.21.

Proposition 4.6. *Let* $\gamma > 0$ *and* $\boldsymbol{x}_0, \boldsymbol{z}_0 \in \mathbb{R}^n$*. Consider the sequence of updates* $\{(\boldsymbol{v}_k, \boldsymbol{x}_k, \boldsymbol{z}_k)\}_{k \geqslant 1}$ *generated by* (1.12)*. Define the sequence* $\{\boldsymbol{u}_k\}_{k \geqslant 1}$ *to be* $\boldsymbol{u}_1 = \boldsymbol{x}_1 + \boldsymbol{z}_1$*, and* $\boldsymbol{u}_{k+1} = \mathrm{T}_{\mathrm{ADMM}}(\boldsymbol{u}_k)$ *for* $k \geqslant 1$*. Then* $\boldsymbol{x}_k = \mathrm{Prox}_{\gamma f}(\boldsymbol{u}_k)$ *for* $k \geqslant 2$*.*

In particular, since $\mathrm{Prox}_{\gamma f}$ is a continuous operator, convergence of $\{\boldsymbol{x}_k\}$ can be established by showing that the sequence $\{\boldsymbol{u}_k\}$ in Proposition 4.6 is convergent. Notice that if the sequence $\{\boldsymbol{x}_k\}$ generated by the equation $\boldsymbol{x}_{k+1} = \mathrm{T}(\boldsymbol{x}_k)$ converges to $\boldsymbol{x}^*$ and if T is continuous (e.g. T is contractive or averaged), then the limit point $\boldsymbol{x}^*$ is necessarily a fixed-point of T, i.e., $\boldsymbol{x}^* = \mathrm{T}(\boldsymbol{x}^*)$. Recall from Chapter 1 that this is referred as fixed-point convergence [175] and the set of fixed points of T is denoted by fix(T). We record the following characterization of fix(T) when T is $\mathrm{T}_{\mathrm{ISTA}}$ or $\mathrm{T}_{\mathrm{ADMM}}$ [175].

Lemma 4.7. *Let* $\mathscr{X}$ *be the minimizers of problem* (1.6)*. Then*

(a) $\boldsymbol{x}^* \in \mathscr{X} \iff \boldsymbol{x}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}})$.

(b) $\boldsymbol{x}^* \in \mathscr{X} \iff \boldsymbol{x}^* = \mathrm{Prox}_{\gamma f}(\boldsymbol{u}^*)$, $\boldsymbol{u}^* \in \mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})$.

In particular, $\mathscr{X}$ is nonempty if and only if $\mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}})$ and $\mathrm{fix}(\mathrm{T}_{\mathrm{ADMM}})$ are nonempty.

Once we have shown that T is contractive or averaged, the convergence of $\{\boldsymbol{x}_k\}$ can be established using the following result.

Theorem 4.8. *Let* T *be an operator on* $\mathbb{R}^n$ *and let* $\boldsymbol{x}_0 \in \mathbb{R}^n$. *Consider the sequence* $\{\boldsymbol{x}_k\}_{k\geqslant 0}$ *generated using* $\boldsymbol{x}_{k+1} = \mathrm{T}(\boldsymbol{x}_k)$. *Then*

(a) If T *is contractive, then* fix(T) *is a singleton, say* $\boldsymbol{x}^*$, *and* $\{\boldsymbol{x}_k\}$ *converges linearly to* $\boldsymbol{x}^*$.

(b) If T *is averaged and* fix(T) *is nonempty, then there exists* $\boldsymbol{x}^* \in \mathrm{fix}(\mathrm{T})$ *such that* $\{\boldsymbol{x}_k\}$ *converges sublinearly to* $\boldsymbol{x}^*$.

Part (a) is simply the classical contraction mapping theorem [175]. Convergence in Part (b) is showed in Lemma 2.6 and convergence rate is derived in [114, 175]. We make a couple of important distinctions between the two results. First, fix(T) is guaranteed to be nonempty when T is a contraction, whereas we need to assume this if T is averaged. Second, when T is a contraction, the sequence converges to the same unique point regardless of the initialization $\boldsymbol{x}_0$, whereas the limit point $\boldsymbol{x}^*$ depends on $\boldsymbol{x}_0$ if T is averaged.

Iterate convergence for proximal algorithms such as ISTA and ADMM follows from the previous results. For ISTA, we assume that f is smooth and convex, and we just assume that f is convex for ADMM. In either case, the regularizer g is assumed to be closed, proper, and convex, and we assume that problem (1.6) is solvable.

Theorem 4.9. *Let* f *and* g *be closed, proper and convex and assume that problem* (1.6) *is solvable. Then*

(a) If ∇f *is* β*-Lipschitz, then the ISTA iterates* (1.10) *converge sublinearly to* $\mathscr{X}$, *for any* $\boldsymbol{x}_0 \in \mathbb{R}^n$ *and* $0 < \gamma < 2/\beta$. *Moreover, if* f *is* μ*-strongly convex and* $0 < \gamma < 2/\max(\beta, \mu)$, *then the convergence is linear.*

(b) The ADMM iterates (1.12) *converge sublinearly to* $\mathscr{X}$, *for any* $\boldsymbol{x}_0 \in \mathbb{R}^n$ *and* $0 < \gamma < 2/\beta$.

Proof. Since $\mathscr{X}$ is nonempty, if follows from Lemma 4.7 that $\mathrm{fix}(\mathrm{T}_{\mathrm{ISTA}}) = \mathscr{X}$. Now, since f is convex, ∇f is β-Lipschitz, and $0 < \gamma < 2/\beta$, it follows from Lemma 4.2 that $\mathrm{I} - \gamma\nabla f$ is averaged. Also, $\mathrm{Prox}_{\gamma g}$ is averaged by Lemma 4.3. Being the composition of $\mathrm{I} - \gamma\nabla f$ and $\mathrm{Prox}_{\gamma g}$, we can conclude from Lemma 4.4 that $\mathrm{T}_{\mathrm{ISTA}}$ is averaged. Hence, we have from Theorem 4.8 that the ISTA iterates converge sublinearly to some $\boldsymbol{x}^* \in \mathscr{X}$.

To establish linear convergence using Theorem 4.8, we show that T_{ISTA} is contractive. Indeed, since f is μ-strongly convex, ∇f is β-Lipschitz, and $0 < \gamma < 2/\max(\beta, \mu)$, it follows from Lemma 4.2 that $I - \gamma \nabla f$ and hence T_{ISTA} is contractive. This proves part (a).

For part (b), since f and g are closed, proper and convex, it follows from Lemma 4.3 that $\text{Prox}_{\gamma f}$ and $\text{Prox}_{\gamma g}$ are 1/2-averaged. Thus, the reflected proximal operators $R_{\gamma f}$ and $R_{\gamma g}$ in (4.2) are nonexpansive. But this means that T_{ADMM} is 1/2-averaged. Moreover, since $\mathscr{X} \neq \phi$ by assumption, it follows from Lemma 4.7 that $\text{fix}(T_{ADMM})$ is nonempty. Thus, we can conclude from Theorem 4.8 that the ADMM iterates converge sublinearly to $\mathscr{X}$. □

The convergence results for ISTA in (1.10) and ADMM in (1.12) in Theorem 4.9 can be extended to PnP-ISTA in (2.24) and PnP-ADMM in (2.25), where $\text{Prox}_{\gamma g}$ is replaced by a denoising operator D. We just need to ensure that D is contractive or averaged. Indeed, the proof of Theorem 4.9 goes through in this case. More precisely, PnP-ISTA converges if D is θ-averaged where $\theta \in (0,1)$ as shown in Theorem 2.8, and PnP-ADMM converges if D is θ-averaged where $\theta \in (0, 1/2]$ as proved in Thorem 2.9. We note that similar convergence results are also available (for nonexpansive D) for RED algorithms [91, 97, 115], PnP-ISTA for compressive sensing [106] and a variant of PnP-ISTA [92].

4.4 Averaged Deep Denoiser

Motivated by the convergence theory of averaged denoisers, we propose a mechanism to train denoisers that are contractive or averaged by construction and whose denoising capability is close to that of modern denoisers. As explained in Section 4.1, training a nonexpansive CNN denoiser presents several challenges. This led us to consider a different solution to the problem based on unfolded networks. More specifically, we unfold ISTA and ADMM operators applied to classical wavelet denoising, which are used to construct trainable patch denoisers that are provably averaged (or contractive). This trained patch denoiser is then used to design an end-to-end image denoiser by using a patch aggregation formula. The novelty in this regard is that we prove that the image denoiser inherits the averaged (or contractive) property of the original patch denoiser.

4.4.1 Wavelet Thresholding and Unfolding

Given a noisy patch $y \in \mathbb{R}^p$, consider the wavelet denoising problem

$$\min_x \psi(x, y) + g(x), \tag{4.3}$$

where $\psi(x, y) = (1/2)||x - y||^2$ and g is the form

$$g(x) = ||\mathbf{H}x||_\Lambda := \sum_{j=1}^{p} \lambda_j |(\mathbf{H}x)_j|,$$

where $\mathbf{H} \in \mathbb{R}^{p\times p}$ is an orthogonal wavelet transform and $\mathbf{\Lambda} = \operatorname{diag}(\lambda_1, \ldots, \lambda_p)$ is a diagonal positive definite matrix. By definition, the closed-form solution of (4.3) is given by the proximal operator of g, namely,

$$\operatorname{Prox}_{\gamma g}(\boldsymbol{x}) = \mathbf{H}^\top \mathrm{S}_{\mathbf{\Lambda}}(\mathbf{H}\boldsymbol{x}), \tag{4.4}$$

where γ is absorbed into Λ and $\mathrm{S}_{\mathbf{\Lambda}} : \mathbb{R}^p \to \mathbb{R}^p$ is the soft-thresholding operator:

$$\forall\, i = 1, \ldots, p: \quad (\mathrm{S}_{\mathbf{\Lambda}}(\boldsymbol{x}))_i = \operatorname{sgn}(\boldsymbol{x}_i)\max(0, |\boldsymbol{x}_i| - \lambda_i).$$

Our present interest is using this closed-form solution in the iterative ISTA and ADMM algorithms for solving (4.3), which we wish to unfold. For example, the ISTA updates are given by $\boldsymbol{x}_{k+1} = \mathrm{T}_{\mathrm{ISTA}}(\boldsymbol{x}_k)$, where $\mathrm{T}_{\mathrm{ISTA}}$ is given by (4.1) but with the difference that the loss term f in (4.1) is now ψ. In particular, substituting $\nabla_x \psi(\boldsymbol{x}, \boldsymbol{y}) = (1-\gamma)\boldsymbol{x} + \gamma\boldsymbol{y}$ and (4.4) in $\mathrm{T}_{\mathrm{ISTA}}$, we obtain the ISTA block.

Definition 4.10. *The ISTA block is the operator* $\mathbf{U} : \mathbb{R}^n \to \mathbb{R}^n$ *given by*

$$\mathbf{U}(\boldsymbol{x}; \gamma, \mathbf{\Lambda}) = \mathbf{H}^\top \mathrm{S}_{\mathbf{\Lambda}}\big(\mathbf{H}\big((1-\gamma)\boldsymbol{x} + \gamma\boldsymbol{y}\big)\big). \tag{4.5}$$

The ISTA block can be interpreted as performing an inversion step using gradient descent on the loss ψ, followed by wavelet regularization via the proximal operator of g. The trainable parameters in (4.5) are γ and $\mathbf{\Lambda}$, and the fixed parameters are the wavelet transform $\mathbf{H}$ and input $\boldsymbol{y}$. Unlike the unfolding in LISTA [48], we use a fixed transform $\mathbf{H}$. This is because we wish to retain the proximal structure in (4.5) and the orthogonality of $\mathbf{H}$ plays a crucial role in this regard. It is difficult to train $\mathbf{H}$ to satisfy orthogonality constraints.

Similar to the ISTA block, we can unfold the ADMM iterations for solving (4.3). For this, we need to compute $\mathrm{R}_{\gamma g}$ and $\mathrm{R}_{\gamma\psi}$ in (4.2). Now, we can express $\operatorname{Prox}_{\gamma g}$ as (4.4), where γ is absorbed in the tunable parameter $\mathbf{\Lambda}$. On the other hand,

$$\operatorname{Prox}_{\gamma\psi}(\boldsymbol{x}) = (1+\gamma)^{-1}(\gamma\boldsymbol{x} + \boldsymbol{y}). \tag{4.6}$$

Substituting (4.4) and (4.6) in (4.2), we obtain the ADMM block.

Definition 4.11. *The ADMM block is the operator* $\mathbf{U} : \mathbb{R}^n \to \mathbb{R}^n$ *given by*

$$\mathbf{U}(\boldsymbol{x}; \gamma, \mathbf{\Lambda}) = \frac{1}{2}\boldsymbol{x} + \frac{1}{2}\Big(2\mathbf{H}^\top \mathrm{S}_{\mathbf{\Lambda}}\big(\mathbf{H}\mathrm{R}_{\gamma\psi}(\boldsymbol{x})\big) - \mathrm{R}_{\gamma\psi}(\boldsymbol{x})\Big), \tag{4.7}$$

where

$$\mathrm{R}_{\gamma\psi}(\boldsymbol{x}) = 2(1+\gamma)^{-1}(\gamma\boldsymbol{x} + \boldsymbol{y}) - \boldsymbol{x}.$$

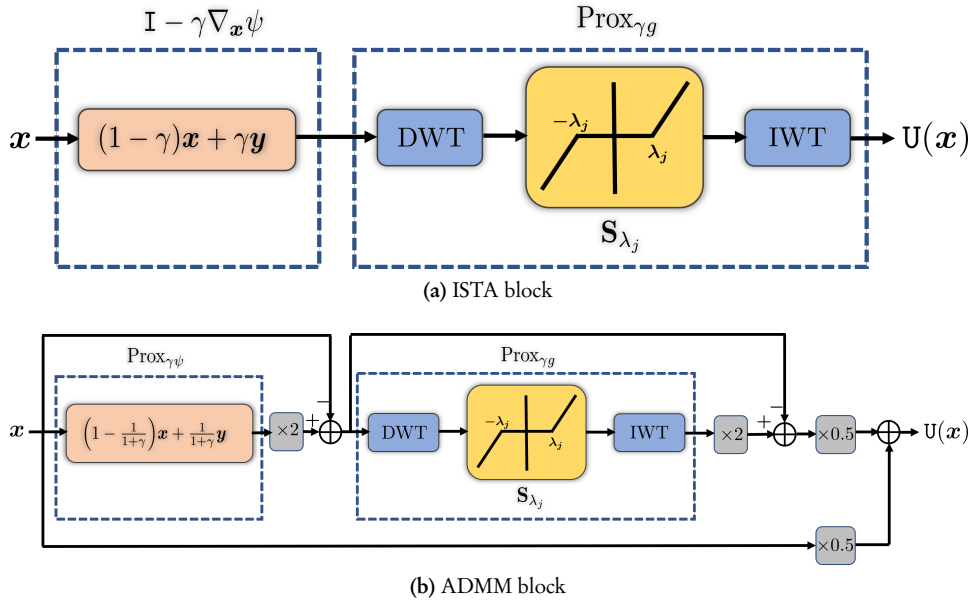


Figure 4.2: Building blocks of the proposed denoiser obtained by unfolding ISTA and ADMM for wavelet denoising.

The ADMM block can be interpreted as performing an inversion by using the proximal operator of ψ, followed by regularization using the proximal operator of g. As with the ISTA block, the trainable parameters in (4.7) are γ and $\mathbf{\Lambda}$, while $\mathbf{H}$ and $\boldsymbol{y}$ are fixed.

The ISTA and ADMM blocks are depicted in Fig. 4.2. We next show that they are averaged by construction; in fact, since ψ is strongly convex in $\boldsymbol{x}$, we can say more for the ISTA block.

Proposition 4.12. *Let $\mathbf{\Lambda}$ be a diagonal positive definite matrix and $\mathbf{H}$ be an orthogonal (wavelet) transform. Then*

(a) The ISTA block (4.5) *is contractive if* $0<\gamma\leqslant 1$.

(b) The ADMM block (4.7) *is* $1/2$*-averaged if* $\gamma>0$.

Proof. For part (a), since $\mathrm{Prox}_{\gamma g}$ is nonexpansive by Lemma 4.3, it suffices to show that $\mathtt{I}-\gamma\nabla_{\boldsymbol{x}}\psi$ is contractive since the composition of a contractive and a nonexpansive operator is contractive. However, for any $\boldsymbol{x}_1,\boldsymbol{x}_2\in\mathbb{R}^n$,

$$\|(\mathtt{I}-\gamma\nabla_{\boldsymbol{x}}\psi)\boldsymbol{x}_1-(\mathtt{I}-\gamma\nabla_{\boldsymbol{x}}\psi)\boldsymbol{x}_2\|=(1-\gamma)\|\boldsymbol{x}_1-\boldsymbol{x}_2\|.$$

Thus, $\mathtt{I}-\gamma\nabla_{\boldsymbol{x}}\psi$ is contractive if $0<\gamma\leqslant 1$.

For part (b), we note that since f and g are convex functions and $\gamma>0$, $\mathrm{Prox}_{\gamma\psi}$ and $\mathrm{Prox}_{\gamma g}$ are $1/2$-averaged by Lemma 4.3. In other words, the reflected proximal operators $\mathrm{R}_{\gamma\psi}$ and $\mathrm{R}_{\gamma g}$ are nonexpansive and so is their composition $\mathrm{R}_{\gamma\psi}\circ\mathrm{R}_{\gamma g}$. Thus, (4.2) and hence the ADMM block in (4.7) is $1/2$-averaged. □

4.4.2 Patch Denoiser

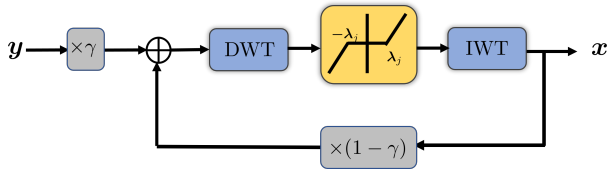


Figure 4.3: ISTA algorithm for Gaussian denoising

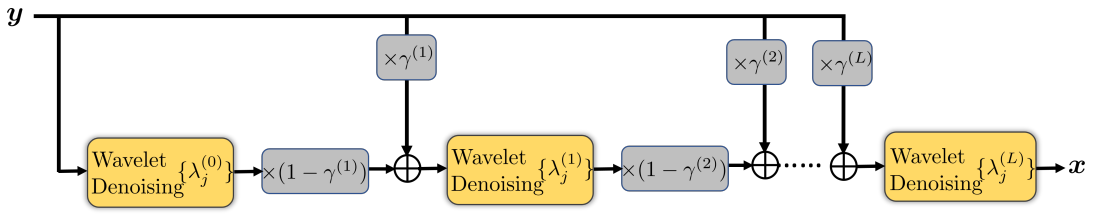


Figure 4.4: Unfolding ISTA algorithm for Gaussian denoising

The proposed patch denoiser $\mathsf{D}_{\text{patch}} : \mathbb{R}^p \to \mathbb{R}^p$ is obtained by concatenation of ISTA or ADMM blocks. That is,

$$\mathsf{D}_{\text{patch}} = \mathbf{U}^{(L)} \circ \mathbf{U}^{(L-1)} \circ \cdots \circ \mathbf{U}^{(1)}, \tag{4.8}$$

where $\mathbf{U}^{(\ell)} = \mathbf{U}(\cdot; \gamma^{(\ell)}, \mathbf{\Lambda}^{(\ell)})$, $\ell = 1, 2, \ldots L$, is either (4.5) or (4.7). It is understood that a fixed orthogonal wavelet transform $\mathbf{H}^{(\ell)}$ is used for each block. In other words, (4.8) is a L-layer end-to-end trainable deep network, where the trainable parameters are $\gamma^{(1)}, \ldots, \gamma^{(L)}$ and $\mathbf{\Lambda}^{(1)}, \ldots, \mathbf{\Lambda}^{(L)}$. Note that the input to (4.8) is a noisy patch represented as a vector of size $p = k^2$ where k is the patch width; the output is a denoised patch of the same size.

Since $\mathsf{D}_{\text{patch}}$ is the composition of averaged (contractive) operators, it is immediate from Lemma 4.4 that $\mathsf{D}_{\text{patch}}$ is averaged (contractive). The precise results are as follows.

Proposition 4.13. *Let $\mathbf{\Lambda}^{(1)}, \ldots, \mathbf{\Lambda}^{(L)}$ be diagonal positive definite and $\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(L)}$ be orthogonal. Then*

(a) D_{patch} is contractive if $0 < \gamma^{(\ell)} \leqslant 1$ and $\mathbf{U}^{(\ell)}$ is (4.5).

(b) D_{patch} is $L/(L+1)$-averaged if $\gamma^{(\ell)} \geqslant 0$ and $\mathbf{U}^{(\ell)}$ is (4.7).

The entire idea of unfolding ISTA iteration to form a patch denoiser is shown in Figures 4.3 and 4.4.

4.4.3 Patch Aggregation

The denoiser $\mathsf{D}_{\text{patch}}$ in the previous section can be used to denoise patches from the noisy image, which would then have to be aggregated to obtain the denoised image [35, 70]. In particular, for patch aggregation, we extract overlapping patches using some stride, denoise

the patches using denoisers in (15) and for overlapped pixels between patches, average them. The challenge is to ensure that the final image-to-image denoiser is averaged or contractive. We next explain in detail how this can be done.

Let $\Omega = [0, q-1]^2$ denote the image domain and $\mathbf{X} : \Omega \to \mathbb{R}$ be the input (noisy) image. We assume periodic (or circular) boundary condition, i.e., for $\boldsymbol{i} = (i_1, i_2) \in \Omega$ and $\tau = (\tau_1, \tau_2) \in \mathbb{Z}^2$, $\boldsymbol{i} - \tau$ is defined as $\big((i_1 - \tau_1) \bmod q, (i_2 - \tau_2) \bmod q\big)$. Without loss of generality, we assume that q is multiple of the patch size k, since we can always zero pad the image to satisfy the condition.

For $\boldsymbol{i} \in \Omega$, let $\mathrm{P}_{\boldsymbol{i}}(\mathbf{X})$ be the vectorized representation of the patch starting at pixel $\boldsymbol{i}$. In particular, $\mathrm{P}_{\boldsymbol{i}}(\mathbf{X}) \in \mathbb{R}^{k^2}$ is obtained by vectorizing the samples $\{\mathbf{X}(\boldsymbol{i} + \tau), \tau \in [0, k-1]^2\}$ in some fixed order. Suppose we choose to denoise patches with a stride s, where s divides k (and hence q). For example, $s = 1$ corresponds to denoising every image patch. Let $q_s = q/s$ and define $\mathbf{J} = \{0, s, 2s, \ldots, (q_s - 1)s\} \times \{0, s, 2s, \ldots, (q_s - 1)s\} \subseteq \Omega$. The points in $\mathbf{J}$ are the starting coordinates of the extracted patches. We extract the patches $\{\mathrm{P}_{\boldsymbol{j}}(\mathbf{X}), \boldsymbol{j} \in \mathbf{J}\}$ from $\mathbf{X}$ with stride s, denoise them using (4.8), and finally average them to obtain the denoised image. More precisely, we define the image-to-image denoiser to be

$$\mathrm{D}(\mathbf{X}) = \frac{1}{k_s^2} \sum_{\boldsymbol{j} \in \mathbf{J}} \mathrm{P}_{\boldsymbol{j}}^{\top}\big(\mathrm{D}_{\text{patch}}(\mathrm{P}_{\boldsymbol{j}}(\mathbf{X}))\big), \tag{4.9}$$

where $k_s = k/s$ and $\mathrm{P}_{\boldsymbol{j}}^{\top}$ is the adjoint of the linear operator $\mathrm{P}_{\boldsymbol{i}}$. The factor k_s^2 is the number of patches that contain a given pixel; the assumption that s divides k is essential to ensure that this is the same for every pixel. Given a vector $\boldsymbol{z} \in \mathbb{R}^p$, $\mathrm{P}_{\boldsymbol{i}}^{\top}(\boldsymbol{z})$ is an image with the same domain Ω as the original image, where the vectorization of the patch at position $\boldsymbol{i}$ is $\boldsymbol{z}$ and the remaining pixels are set to zero. A precise definition of $\mathrm{P}_{\boldsymbol{i}}^{\top}$ is provided in the Appendix.

When ISTA blocks are used in $\mathrm{D}_{\text{patch}}$, we denote the image denoiser in (4.9) as D_{ISTA} and when ADMM blocks are used, we denote (4.9) as D_{ADMM}. We have the following result for D_{ISTA} and D_{ADMM}.

Proposition 4.14. *If* D_{patch} *is contractive, then* D *is contractive, and if* D_{patch} *is* θ*-averaged, then* D *is* θ*-averaged. In particular,* D_{ISTA} *is contractive and* D_{ADMM} *is* $L/(L+1)$*-averaged.*

Detailed proof can be found in the Appendix. The core idea is to express D as the average of contractive or averaged operators and apply Lemma 4.5.

Since contractive and averaged operators are nonexpansive, the proposed denoiser can be used to guarantee convergence of an iterative algorithm that requires the denoiser to be contractive, averaged, or nonexpansive. In particular, we record the following important implication of the results so far.

Corollary 4.15. *Suppose the loss term f in* (1.6) *is convex. Then if we plug* $\mathrm{D}_{\mathrm{ISTA}}$ *or* $\mathrm{D}_{\mathrm{ADMM}}$ *into PnP-ISTA in* (1.20)*, RED-PG [92] and algorithms such as [81, 97, 82, 115], we can guarantee iterate convergence of the corresponding algorithm.*

4.4.4 Training Details

We train our denoiser $\mathrm{D}_{\mathrm{patch}}$ on $k \times k$ patches extracted from images of the BSD500 dataset. Denote the clean patches (normalized to $[0,1]$ intensity) by $\boldsymbol{z}_1,\ldots,\boldsymbol{z}_N \in \mathbb{R}^p$, where $p = k^2$. The noisy images $\hat{\boldsymbol{z}}_1,\ldots,\hat{\boldsymbol{z}}_N \in \mathbb{R}^p$ are generated by corrupting $\boldsymbol{z}_i$ with additive white Gaussian noise, that is, we set $\hat{\boldsymbol{z}}_i = \boldsymbol{z}_i + \sigma\boldsymbol{\eta}_i$, where $\boldsymbol{\eta}_i \sim \mathcal{N}(0,\mathrm{I})$ and $\sigma \in [0,1]$ is the noise level. We wish to learn the parameters of $\mathrm{D}_{\mathrm{patch}} = \mathrm{D}_{\mathrm{patch}}(\cdot;\Theta)$ such that $\boldsymbol{z}_i \approx \mathrm{D}_{\mathrm{patch}}(\hat{\boldsymbol{z}}_i;\Theta)$, where

$$\Theta = \left\{ \gamma^{(\ell)}, \lambda_j^{(\ell)} : \ell = 1,2,\ldots L,\ j = 1,2,\ldots p \right\}$$

are the trainable parameters. In view of the Proposition 4.12, when each $\mathbf{U}^{(\ell)}$ is the ISTA block, the denoiser is trained via the following optimization:

$$\begin{aligned} \min_{\Theta} \quad & \frac{1}{N}\sum_{i=1}^{N} \|\mathrm{D}_{\mathrm{patch}}(\hat{\boldsymbol{z}}_i;\Theta) - \boldsymbol{z}_i\|^2 \\ \text{s.t.} \quad & \gamma^{(1)},\ldots,\gamma^{(L)} \in [0,1], \\ & \lambda_j^{(1)},\ldots,\lambda_j^{(L)} \in [0,\infty) \quad (j = 1,2,\ldots p). \end{aligned} \tag{4.10}$$

When each $\mathbf{U}^{(\ell)}$ is the ADMM block, the denoiser is trained via the following optimization:

$$\begin{aligned} \min_{\Theta} \quad & \frac{1}{N}\sum_{i=1}^{N} \|\mathrm{D}_{\mathrm{patch}}(\hat{\boldsymbol{z}}_i;\Theta) - \boldsymbol{z}_i\|^2 \\ \text{s.t.} \quad & \gamma^{(1)},\ldots,\gamma^{(L)} \in [0,\infty), \\ & \lambda_j^{(1)},\ldots,\lambda_j^{(L)} \in [0,\infty) \quad (j = 1,2,\ldots p). \end{aligned} \tag{4.11}$$

For the experiments reported here, we have set the patch size to $k = 64$ and have used a 10-fold cascade of Daubechies, Haar, and Symlet orthogonal wavelets in succession (i.e. $L = 30$). We solve (4.10) and (4.11) using Adam optimizer [144]. We use mini-batches of size 128 to run the optimizer only for 5 epochs. After every gradient step (performed using automatic differentiation) on Θ, we enforce the constraints in (4.10) and (4.11) simply by projecting the variables onto the intervals in (4.10) and (4.11). In other words, we use a form of projected gradient descent. We separately pre-train denoisers over the noise range 5-50 in intervals of 5. Our trained models are light-weight requiring about 500 kB for storage, as compared to DnCNN which takes around 2.3 MB. While inferring the denoisers, the stride value is set to 8 for all experiments.

4.5 Experiments

4.5.1 Empirical Analysis

In this section, we analyze the proposed denoisers and compare their reconstruction accuracy and convergence aspects with existing denoisers.

Experiment 1: We start by presenting numerical evidence which show that some popular denoisers—BM3D [70] and pre-trained networks DnCNN [9] and N-CNN [78]—are not nonexpansive, especially, if we restrict the input to images encountered in the PnP pipeline. More precisely, we consider PnP-ISTA applied to image superresolution, and compute the output-input ratio $||\boldsymbol{x}_{k+1}-\boldsymbol{x}_k||/||\boldsymbol{z}_{k+1}-\boldsymbol{z}_k||$ for the denoising step $\boldsymbol{x}_{k+1}=\mathrm{D}(\boldsymbol{z}_{k+1})$ in (1.20). This gives a lower bound for the Lipschitz constant of the plugged denoiser—if any of these ratios is >1, then the Lipschitz constant cannot be less than 1, i.e., the denoiser cannot be nonexpansive. The ratio for different denoisers across the iterations is shown in Fig. 4.1. We notice that the ratio is >1 at many iterations for BM3D, DnCNN, and N-CNN, proving that these denoisers are not nonexpansive. For N-CNN, the residual is constrained to be nonexpansive, but this constraint is also violated as shown in Fig. 4.1(c). On the other hand, the ratio is always <1 for our denoiser, which is consistent with the theoretical results in Section 4.4.

Experiment 2: We now give an example in which the PnP iterates are shown to diverge if we plug DnCNN [9] or N-CNN [78]. We have already seen that these denoisers are not nonexpansive, hence it is indeed possible for the iterates to diverge. In particular, we perform image deblurring using PnP-ISTA and show that the distance between successive iterates $||\boldsymbol{x}_k-\boldsymbol{x}_{k-1}||$ diverges, which means that the sequence $(\boldsymbol{x}_k)$ does not converge. We have used $\gamma=1.95$ for DnCNN and $\gamma=0.001$ for N-CNN, which are in agreement with the convergence results in Theorem 2.8. On the other hand, for the same experiment and parameter settings, we notice that $||\boldsymbol{x}_k-\boldsymbol{x}_{k-1}||$ is strictly decreasing for the proposed denoisers, thanks to their averaged property.

Experiment 3: Next, we compare the denoising performance of $\mathrm{D}_{\mathrm{ISTA}}$ and $\mathrm{D}_{\mathrm{ADMM}}$ with BM3D, DnCNN and N-CNN. The results are shown in Table 4.3. Our denoisers are off by 0.9-1.0 dB from BM3D and by 1.4-1.6 dB from DnCNN, This is possibly due to our choice of wavelet-based regularizer as the main regularization block in $\mathrm{D}_{\mathrm{ISTA}}$ and $\mathrm{D}_{\mathrm{ADMM}}$. However, $\mathrm{D}_{\mathrm{ISTA}}$ and $\mathrm{D}_{\mathrm{ADMM}}$ outperform N-CNN at all noise levels. Importantly, the reconstruction capability of our denoiser is competitive with the best denoisers as shown in Section 4.5.2. This is not surprising since there is no direct relationship between denoising capacity and the final restoration quality of PnP algorithms. For example, BM3D has been shown to give better restoration results than DnCNN in some applications [78], though DnCNN generally has better denoising capabilities than BM3D.

Table 4.3: Denoising performance for BSD68 dataset.

Method	BM3D	DnCNN	N-CNN	$\mathrm{D}_{\mathrm{ISTA}}$	$\mathrm{D}_{\mathrm{ADMM}}$
$\sigma = 5$	37.52	38.01	33.35	36.54	36.62
$\sigma = 20$	29.65	30.26	27.99	28.63	28.63
$\sigma = 35$	27.10	27.69	25.17	26.19	26.17

Experiment 4: Finally, we present numerical evidence showing convergence for PnP-ISTA using the proposed denoisers. This is done using a deblurring experiment on images from the Set 12 dataset. The results are shown in Fig. 4.5, where we plot $||\boldsymbol{x}_k - \boldsymbol{x}_{k-1}||$, PSNR and lower bound for the Lipschitz constant for different k. As expected, $||\boldsymbol{x}_k - \boldsymbol{x}_{k-1}||$ is strictly decreasing and the lower bound for the Lipschitz constant is < 1. Note that our PSNR stabilizes in less than 50 iterations.

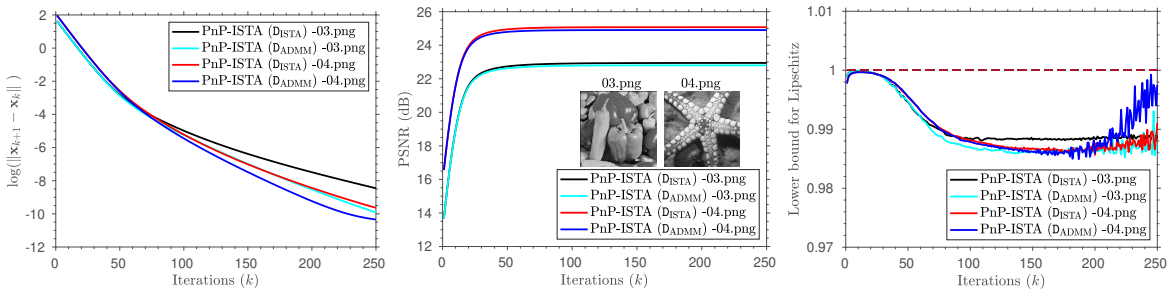


Figure 4.5: Analysis of the proposed denoisers. Denoisers $\mathrm{D}_{\mathrm{ISTA}}$ and $\mathrm{D}_{\mathrm{ADMM}}$ are plugged into PnP-ISTA which is used for image deblurring. We plot the difference between successive iterates (log scale), PSNR (dB), and the lower bound for the Lipschitz constant with iterations. We notice that the difference between iterates is strictly decreasing thanks to the averaged property, the PSNR stabilizes in 50-100 iterations and the lower bound for Lipschitz constant is < 1.

4.5.2 Applications

In this section, we present results for image deblurring and superresolution by plugging our denoiser into PnP-ISTA and RED-PG [92]. We show that our results on the Set 12 dataset are competitive with that obtained using the best denoisers. Importantly, as noted in Corollary 4.15, iterate convergence for PnP-ISTA and RED-PG is guaranteed for $\mathrm{D}_{\mathrm{ISTA}}$ and $\mathrm{D}_{\mathrm{ADMM}}$. Indeed, the updates for RED-PG are given by

$$\begin{aligned} \boldsymbol{z}_{k+1} &= \mathrm{Prox}_{\gamma f}(\boldsymbol{x}_k), \\ \boldsymbol{x}_{k+1} &= \left(1 - \frac{1}{L}\right)\boldsymbol{x}_k + \frac{1}{L}\mathrm{D}(\boldsymbol{z}_{k+1}), \end{aligned} \tag{4.12}$$

where $\gamma > 0$ and $L > 1$ [92]. Thus, we can write $\boldsymbol{x}_{k+1} = \mathrm{T}(\boldsymbol{x}_k)$, where T is averaged being the composition of two averaged operators.

Deblurring: Refer to Section 2.2.4 for the forward model for deblurring. For experiments 1 and 2 in Section 4.5.1 and in particular the results in Fig. 4.5 and Table 4.1, **F** is a Gaussian blur of width 25×25 and standard deviation 1.6 and the noise level is $\sigma = 0.04$. In Fig. 4.6, where **F** is a horizontal motion blur, we have compared $\mathrm{D}_{\mathrm{ISTA}}$ and $\mathrm{D}_{\mathrm{ADMM}}$ with high-performance denoisers for the RED-PG algorithm. BM3D gives the best results for this experiment, however, it is evident from the error images and the quality metrics, that the results obtained using our denoiser are comparable to BM3D. In Table 4.4, where **F** is one of the blur kernels from [176], we have compared our denoiser with existing trained denoisers on some standard images, again for the RED-PG algorithm. We notice that our performance is comparable with DnCNN.

(a) Input **(b)** BM3D **(c)** DnCNN **(d)** N-CNN **(e)** $\mathrm{D}_{\mathrm{ISTA}}$ **(f)** $\mathrm{D}_{\mathrm{ADMM}}$

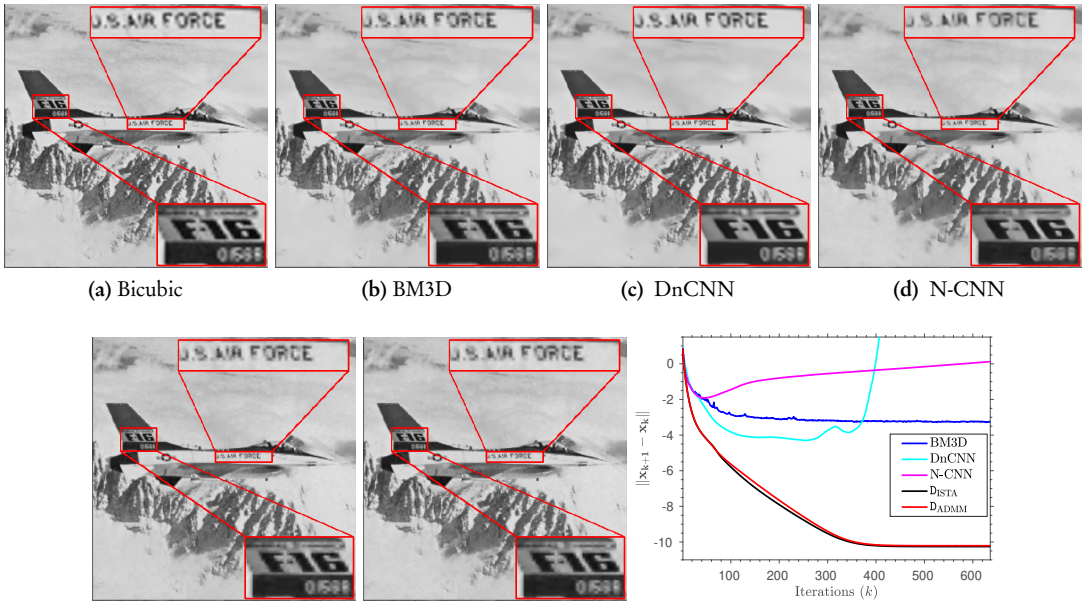
(g) Ground Truth **(h)** Error image (b) - (g) **(i)** Error image (c) - (g) **(j)** Error image (d) - (g) **(k)** Error image (e) - (g) **(l)** Error image (f) - (g)

Figure 4.6: Deblurring using RED-PG with different denoisers. PSNR and SSIM values are: Input (13.43, 0.4142), BM3D (**40.04**, 0.9506), DnCNN (37.75, 0.9141), N-CNN (36.34, 0.9095), $\mathrm{D}_{\mathrm{ISTA}}$ (**40.04**, **0.9570**) and $\mathrm{D}_{\mathrm{ADMM}}$ (39.53, 0.9507). The error images are also shown. The performance of the proposed denoisers is comparable to BM3D and DnCNN.

Superresolution: Refer to Section 2.2.4 for the forward model for superresolution. For Experiment 2 and Table 4.2 in Section 4.5.1, **B** is a Gaussian blur of size 25×25 and standard deviation 1.6, and the noise level 0.04. In Fig. 4.7, where **B** is the identity operator and the noise level is $5/255$, we have compared our denoisers with BM3D, DnCNN, and N-CNN for the PnP-ISTA algorithm. It is clear from the visual results and the metrics that the restoration results using $\mathrm{D}_{\mathrm{ISTA}}$ and $\mathrm{D}_{\mathrm{ADMM}}$ are comparable with BM3D and DnCNN. This is possible only because we train the shrinkage parameters in the wavelet denoising module in the denoisers $\mathrm{D}_{\mathrm{ISTA}}$ and $\mathrm{D}_{\mathrm{ADMM}}$. We also report the values of $||\boldsymbol{x}_k - \boldsymbol{x}_{k-1}||$ for sufficiently large k. This monotonically goes to zero for our denoisers as expected, but is seen to diverge for DnCNN and N-CNN.

Table 4.4: Comparison of deblurring performance for the kernel in [176].

	DnCNN	N-CNN	D_{ISTA}	D_{ADMM}
barbara.png	**25.08**/0.7307	23.98/0.6892	**25.08**/**0.7410**	24.98/0.7371
ship.png	27.39/0.7502	26.82/0.7145	**27.52**/**0.7627**	27.47/0.7600
cameraman.png	24.92/0.7867	**25.01**/0.7700	24.96/**0.7808**	24.91/**0.7808**
couple.png	26.95/0.7346	26.50/0.7046	**27.15**/**0.7541**	27.09/0.7508
fingerprint.png	25.86/0.8773	23.55/0.7817	**25.99**/**0.8843**	25.90/0.8809
hill.png	28.41/0.7301	28.19/0.7125	**28.71**/**0.7573**	28.64/0.7532
house.png	**30.29**/**0.8310**	29.85/0.8252	29.92/0.8269	29.90/0.8278
lena.png	**30.78**/**0.8547**	30.36/0.8421	30.73/0.8540	30.68/0.8545
man.png	28.20/0.7749	27.94/0.7550	**28.45**/**0.7928**	28.38/0.7902
montage.png	24.56/**0.8881**	25.10/0.8786	**24.78**/0.8653	24.70/0.8683
peppers.png	**24.96**/0.8227	24.95/0.8174	24.95/**0.8255**	24.90/0.8252



(a) Bicubic (b) BM3D (c) DnCNN (d) N-CNN

(e) D_{ISTA} (f) D_{ADMM} (g) Convergence plot

Figure 4.7: Image superresolution using PnP-ISTA with different denoisers. PSNR and SSIM after 30 iterations: BM3D (**31.03**, **0.9250**), DnCNN (30.22, 0.9167), N-CNN (28.92, 0.8925), D_{ISTA} (30.61, 0.9167) and D_{ADMM} (30.48, **0.9202**). Reconstructions using the proposed denoisers D_{ISTA} and D_{ADMM} are comparable with BM3D and DnCNN. As shown in the convergence plot, the PnP iterates diverge for DnCNN and N-CNN.

In summary, the proposed denoisers $\mathrm{D}_{\mathrm{ISTA}}$ and $\mathrm{D}_{\mathrm{ADMM}}$ behave as per the predictions in Proposition 4.13 and Corollary 4.15 and their restoration capacity is found to be competitive with BM3D and DnCNN for deblurring and superresolution.

4.6 Conclusion

We trained averaged (contractive) patch denoisers by unfolding ISTA and ADMM applied to wavelet denoising, and showed how they can be extended for image denoising using patch aggregation while preserving their averaged (contractive) property. Moreover, we showed that their regularization capacity for PnP and RED could be brought to par with BM3D and DnCNN by adding sufficiently many layers. Unlike existing CNN denoisers, we can guarantee convergence of PnP and RED using the proposed denoisers and this was verified numerically. In particular, our contribution is an algorithm paradigm to exploit the properties of proximal operators in unfolded algorithms to come up with averaged and contractive denoisers. Though the learning-based denoiser in this chapter is nonexpansive, the denoising performance is off to DnCNN by 1-1.5 dB, and hence there might be applications where DnCNN has better regularization performance than our proposed averaged denoiser. Note that this was never the case in our experiments.

4.7 Appendix

4.7.1 Proof of Lemma 4.2(b)

Proposition 4.16. *If $f : \mathbb{R}^n \to \mathbb{R}$ is convex and β-smooth, then the following statements are equivalent,*

$$
\begin{aligned}
&a)\ \frac{\beta}{2}\boldsymbol{x}^\top\boldsymbol{x} - f(\boldsymbol{x}) \text{ is convex.}\\
&b)\ \langle \nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle \leqslant \beta \|\boldsymbol{x}-\boldsymbol{y}\|^2\\
&c)\ \langle \nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle \geqslant \frac{1}{\beta}\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\|^2\\
&d)\ \|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \leqslant \beta\, \|\boldsymbol{x}-\boldsymbol{y}\|
\end{aligned}
$$

Proposition 4.17. *If $f : \mathbb{R}^n \to \mathbb{R}$ is μ- strongly convex and β-smooth, then $\beta \geqslant \mu$ and the following statements hold,*

$$
\begin{aligned}
&a)\ f(\boldsymbol{x}) - \frac{\mu}{2}\boldsymbol{x}^\top\boldsymbol{x} \text{ is convex.}\\
&b)\ \langle \nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle \geqslant \mu \|\boldsymbol{x}-\boldsymbol{y}\|^2
\end{aligned}
$$

We refer for the proof of Proposition 4.16 and for the proof of Proposition 4.17.

We will prove Lemma 4.2b) by showing for any $\gamma \in \mathbb{R}$ and for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$,

$$||(\mathrm{I}-\gamma\nabla f)(\boldsymbol{x})-(\mathrm{I}-\gamma\nabla f)(\boldsymbol{y})|| \leqslant \max\{|1-\gamma\mu|, |1-\gamma\beta|\}\,||\boldsymbol{x}-\boldsymbol{y}||\,.$$

Define $g(\boldsymbol{x}) := f(\boldsymbol{x}) - \frac{\mu}{2}\boldsymbol{x}^\top\boldsymbol{x}$ and $h(\boldsymbol{x}) := \frac{\beta}{2}\boldsymbol{x}^\top\boldsymbol{x} - f(\boldsymbol{x})$. By hypothesis, Propositions 4.16 and 4.17, g and h are convex. Also $\nabla g(\boldsymbol{x}) = \nabla f(\boldsymbol{x}) - \mu\boldsymbol{x}$ and $\nabla h(\boldsymbol{x}) = \beta\boldsymbol{x} - \nabla f(\boldsymbol{x})$. Thus the gradient operator is given by

$$\nabla f = \nabla g + \mu\mathrm{I} = \beta\mathrm{I} - \nabla h,$$

and hence,

$$\mathrm{I}-\gamma\nabla f = (1-\gamma\mu)\mathrm{I} - \gamma\nabla g = (1-\gamma\beta)\mathrm{I} + \gamma\nabla h, \tag{4.13}$$

$$\begin{aligned}&\langle \nabla g(\boldsymbol{x}) - \nabla g(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle = \langle \nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle - \mu||\boldsymbol{x}-\boldsymbol{y}||^2 \\ &\leqslant (\beta-\mu)||\boldsymbol{x}-\boldsymbol{y}||^2 \text{ by Proposition 4.16 as } f \text{ is } \beta\text{-smooth.}\end{aligned} \tag{4.14}$$

Hence by Proposition 4.16, g is $(\beta-\mu)$-smooth and,

$$||\nabla g(\boldsymbol{x}) - \nabla g(\boldsymbol{y})||^2 \leqslant (\beta-\mu)\,\langle \nabla g(\boldsymbol{x}) - \nabla g(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle\,. \tag{4.15}$$

Similarly,

$$\begin{aligned}&\langle \nabla h(\boldsymbol{x}) - \nabla h(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle = \beta||\boldsymbol{x}-\boldsymbol{y}||^2 - \langle \nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle \\ &\leqslant (\beta-\mu)||\boldsymbol{x}-\boldsymbol{y}||^2 \text{ by Proposition 4.17 as } f \text{ is } \mu\text{-strongly convex.}\end{aligned} \tag{4.16}$$

Hence by Proposition 4.16, h is $(\beta-\mu)$-smooth and,

$$||\nabla h(\boldsymbol{x}) - \nabla h(\boldsymbol{y})||^2 \leqslant (\beta-\mu)\,\langle \nabla h(\boldsymbol{x}) - \nabla h(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle\,. \tag{4.17}$$

Case 1: $\gamma \leqslant 0$

Define $p(\boldsymbol{x}) := \frac{1}{2}\boldsymbol{x}^\top\boldsymbol{x} - \gamma f(\boldsymbol{x})$. Since $-\gamma > 0$, p is convex and differentiable with $\nabla p = \mathrm{I} - \gamma\nabla f$. Hence,

$$\begin{aligned}&\langle \nabla p(\boldsymbol{x}) - \nabla p(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle = ||\boldsymbol{x}-\boldsymbol{y}||^2 - \gamma\,\langle \nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y}), \boldsymbol{x}-\boldsymbol{y}\rangle \\ &\leqslant (1-\gamma\beta)||\boldsymbol{x}-\boldsymbol{y}||^2 \text{ by Proposition 4.16 and since } -\gamma > 0 \\ &\leqslant \max\{1-\gamma\mu, 1-\gamma\beta\}||\boldsymbol{x}-\boldsymbol{y}||^2 \text{ since } 1-\gamma\beta \geqslant 1-\gamma\mu \geqslant 0.\end{aligned}$$

Again, by resorting to Proposition 4.16, the statement is proved.

Case 2: $\gamma \in \left(0, \frac{2}{\beta+\mu}\right]$

$$\begin{aligned}
&\|(\mathrm{I}-\gamma\nabla f)(\boldsymbol{x})-(\mathrm{I}-\gamma\nabla f)(\boldsymbol{y})\|^2 = \|(1-\gamma\beta)(\boldsymbol{x}-\boldsymbol{y})+\gamma(\nabla h(\boldsymbol{x})-\nabla h(\boldsymbol{y}))\|^2 && (4.18)\\
&= (1-\gamma\beta)^2\|\boldsymbol{x}-\boldsymbol{y}\|^2+\gamma^2\|\nabla h(\boldsymbol{x})-\nabla h(\boldsymbol{y})\|^2+2\gamma(1-\gamma\beta)\langle\nabla h(\boldsymbol{x})-\nabla h(\boldsymbol{y}),\boldsymbol{x}-\boldsymbol{y}\rangle\\
&\leqslant (1-\gamma\beta)^2\|\boldsymbol{x}-\boldsymbol{y}\|^2+\left(\gamma^2(\beta-\mu)+2\gamma(1-\gamma\beta)\right)\langle\nabla h(\boldsymbol{x})-\nabla h(\boldsymbol{y}),\boldsymbol{x}-\boldsymbol{y}\rangle. && (4.19)
\end{aligned}$$

The equality (4.18) is because of the operator form in (4.13) and inequality (4.19) holds by (4.17). Note that $\gamma^2(\beta-\mu)+2\gamma(1-\gamma\beta) = -\gamma^2(\beta+\mu)+2\gamma \geqslant 0$ since $\gamma \in \left(0, \frac{2}{\beta+\mu}\right]$. Thus by Proposition (4.16),

$$\left(\gamma^2(\beta-\mu)+2\gamma(1-\gamma\beta)\right)\langle\nabla h(\boldsymbol{x})-\nabla h(\boldsymbol{y}),\boldsymbol{x}-\boldsymbol{y}\rangle \leqslant \left(\gamma^2(\beta-\mu)+2\gamma(1-\gamma\beta)\right)(\beta-\mu)\|\boldsymbol{x}-\boldsymbol{y}\|^2.$$

Note that $\left(\gamma^2(\beta-\mu)+2\gamma(1-\gamma\beta)\right)(\beta-\mu) = (1-\gamma\mu)^2-(1-\gamma\beta)^2 \geqslant 0$. Thus $|1-\gamma\mu| \geqslant |1-\gamma\beta|$ and (4.18) becomes,

$$\begin{aligned}
&\|(\mathrm{I}-\gamma\nabla f)(\boldsymbol{x})-(\mathrm{I}-\gamma\nabla f)(\boldsymbol{y})\|^2 \leqslant (1-\gamma\mu)^2\|\boldsymbol{x}-\boldsymbol{y}\|^2\\
&= \max\{|1-\gamma\mu|,|1-\gamma\beta|\}^2\|\boldsymbol{x}-\boldsymbol{y}\|^2 \text{ since } |1-\gamma\mu| \geqslant |1-\gamma\beta|.
\end{aligned}$$

Hence the statement is proved.

Case 3: $\gamma \in \left(\frac{2}{\beta+\mu}, \infty\right)$

$$\begin{aligned}
&\|(\mathrm{I}-\gamma\nabla f)(\boldsymbol{x})-(\mathrm{I}-\gamma\nabla f)(\boldsymbol{y})\|^2 = \|(1-\gamma\mu)(\boldsymbol{x}-\boldsymbol{y})-\gamma(\nabla g(\boldsymbol{x})-\nabla g(\boldsymbol{y}))\|^2 && (4.20)\\
&= (1-\gamma\mu)^2\|\boldsymbol{x}-\boldsymbol{y}\|^2+\gamma^2\|\nabla g(\boldsymbol{x})-\nabla g(\boldsymbol{y})\|^2-2\gamma(1-\gamma\mu)\langle\nabla g(\boldsymbol{x})-\nabla g(\boldsymbol{y}),\boldsymbol{x}-\boldsymbol{y}\rangle\\
&\leqslant (1-\gamma\mu)^2\|\boldsymbol{x}-\boldsymbol{y}\|^2+\left(\gamma^2(\beta-\mu)-2\gamma(1-\gamma\mu)\right)\langle\nabla g(\boldsymbol{x})-\nabla g(\boldsymbol{y}),\boldsymbol{x}-\boldsymbol{y}\rangle. && (4.21)
\end{aligned}$$

The equality (4.20) is because of the operator form in (4.13) and inequality (4.21) holds by (4.15). Note that $\gamma^2(\beta-\mu)-2\gamma(1-\gamma\mu) = \gamma^2(\beta+\mu)-2\gamma \geqslant 0$ since $\gamma \in \left(\frac{2}{\beta+\mu}, \infty\right)$. Thus by Proposition (4.14),

$$\left(\gamma^2(\beta-\mu)-2\gamma(1-\gamma\mu)\right)\langle\nabla g(\boldsymbol{x})-\nabla g(\boldsymbol{y}),\boldsymbol{x}-\boldsymbol{y}\rangle \leqslant \left(\gamma^2(\beta-\mu)-2\gamma(1-\gamma\mu)\right)(\beta-\mu)\|\boldsymbol{x}-\boldsymbol{y}\|^2.$$

Note that $\left(\gamma^2(\beta-\mu)-2\gamma(1-\gamma\mu)\right)(\beta-\mu) = (1-\gamma\beta)^2-(1-\gamma\mu)^2 \geqslant 0$. Thus $|1-\gamma\beta| \geqslant |1-\gamma\mu|$ and (4.20) becomes,

$$\begin{aligned}
&\|(\mathrm{I}-\gamma\nabla f)(\boldsymbol{x})-(\mathrm{I}-\gamma\nabla f)(\boldsymbol{y})\|^2 \leqslant (1-\gamma\beta)^2\|\boldsymbol{x}-\boldsymbol{y}\|^2\\
&= \max\{|1-\gamma\mu|,|1-\gamma\beta|\}^2\|\boldsymbol{x}-\boldsymbol{y}\|^2 \text{ since } |1-\gamma\beta| \geqslant |1-\gamma\mu|.
\end{aligned}$$

Hence the statement is proved.

4.7.2 Proof of Proposition 4.14

We recall some notations and definitions from Section 4.4. The input image is denoted by $\mathbf{X}:\Omega\to\mathbb{R}$, where $\Omega=[0,q-1]^2$ is the image domain. We also consider the image in matrix form as $\mathbf{X}\in\mathbb{R}^{q\times q}$. We use circular shifts, i.e., for $\boldsymbol{i}=(i_1,i_2)\in\Omega$ and $\tau=(\tau_1,\tau_2)\in\mathbb{Z}^2$, $\boldsymbol{i}-\tau$ is defined as $((i_1-\tau_1) \bmod q,(i_2-\tau_2) \bmod q)$.

For $\boldsymbol{i}\in\Omega$, the patch extraction operator $\mathrm{P}_{\boldsymbol{i}}:\mathbb{R}^{q\times q}\to\mathbb{R}^p$ is a linear operator such that $\mathrm{P}_{\boldsymbol{i}}(\mathbf{X})$ is the vector representation of the samples $\{\mathbf{X}(\boldsymbol{i}+\tau):\tau\in[0,k-1]^2\}$, where $p=k^2$ is the number of pixels in a patch. The adjoint of $\mathrm{P}_{\boldsymbol{i}}$ appears in definition (4.9) and has the following mathematical characterization. Consider the standard inner product on $\mathbb{R}^p$:

$$\forall\,\boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^p:\quad \langle\boldsymbol{x},\boldsymbol{y}\rangle=\boldsymbol{x}^\top\boldsymbol{y},$$

and the Frobenius inner-product on $\mathbb{R}^{q\times q}$:

$$\forall\,\mathbf{X},\mathbf{Y}\in\mathbb{R}^{q\times q}:\quad \langle\mathbf{X},\mathbf{Y}\rangle=\mathrm{Trace}(\mathbf{X}^\top\mathbf{Y}).$$

We will use the same notation $\|\cdot\|$ for the norms induced by these inner products; this should be clear from the context. The adjoint of $\mathrm{P}_{\boldsymbol{i}}$ is the unique linear operator $\mathrm{P}_{\boldsymbol{i}}^\top:\mathbb{R}^p\to\mathbb{R}^{q\times q}$ such that

$$\forall\,\mathbf{X}\in\mathbb{R}^{q\times q},\ \boldsymbol{y}\in\mathbb{R}^p:\quad \langle\boldsymbol{y},\mathrm{P}_{\boldsymbol{i}}(\mathbf{X})\rangle=\left\langle\mathrm{P}_{\boldsymbol{i}}^\top(\boldsymbol{y}),\mathbf{X}\right\rangle,$$

In particular, $\mathrm{P}_{\boldsymbol{i}}^\top$ maps a vector $\boldsymbol{y}\in\mathbb{R}^p$ to a $k\times k$ patch (using the same ordering used to construct $\mathrm{P}_{\boldsymbol{i}}$) and places it in location $\boldsymbol{i}$ and the remaining pixels are assigned zero value. We extract patches with stride s, where s divides k. Let $q_s=q/s$ and $\mathbf{J}=\{0,s,2s,\dots(q_s-1)s\}\times\{0,s,2s,\dots(q_s-1)s\}$; Note that that $|\mathbf{J}|=q_s^2$ is the number of patches and every pixel belongs to k_s^2 patches, where $k_s=k/s$.

The main idea behind Proposition 4.14 is to express the original overlapping patches in terms of multiple sets of non-overlapping patches; this allows us to use some form of orthogonality in the calculations. In particular, let $q_k=q/k$. Note that since we assume q to be a multiple of k, q_k is an integer. We can extract q_k^2 non-overlapping patches from $\mathbf{X}$. More precisely, let

$$\mathbf{J}_0=\big\{0,k,2k,\dots(q_k-1)k\big\}\times\big\{0,k,2k,\dots(q_k-1)k\big\}.$$

By construction, $\mathbf{J}_0\subseteq\mathbf{J}$ and the points in $\mathbf{J}_0$ are the starting coordinates of non-overlapping patches. It is not difficult to see that $\mathbf{J}$ is the disjoint union of (circular) shifts of $\mathbf{J}_0$:

$$\mathbf{J}=\bigcup_{m,n=0}^{k_s-1}\big(\mathbf{J}_0+s(m,n)\big).\tag{4.22}$$

In particular, using (4.22), we can write (4.9) as

$$\mathrm{D} = \frac{1}{k_s^2} \sum_{m,n=0}^{k_s-1} \mathrm{D}_{mn},$$

where D_{mn} is defined as

$$\mathrm{D}_{mn} = \sum_{j \in \mathrm{J}_0} \mathrm{P}_{j+s(m,n)}^{\top} \Big(\mathrm{D}_{\text{patch}}(\mathrm{P}_{j+s(m,n)}) \Big). \tag{4.23}$$

We make the following claims which can be easily verified.

(i) For any $m, n \in \{0, \ldots, k_s - 1\}$,

$$\mathrm{I} = \sum_{j \in \mathrm{J}_0} \mathrm{P}_{j+s(m,n)}^{\top} \mathrm{P}_{j+s(m,n)}, \tag{4.24}$$

where I is the identity operator on $\mathbb{R}^{q \times q}$.

(ii) For any $\mathbf{X} : \Omega \to \mathbb{R}$ and any $m, n \in \{0, \ldots, k_s - 1\}$,

$$\sum_{j \in \mathrm{J}_0} ||\mathrm{P}_{j+s(m,n)}(\mathbf{X})||^2 = ||\mathbf{X}||^2. \tag{4.25}$$

(iii) For any set of vectors $\{\boldsymbol{y}_j \in \mathbb{R}^p : j \in \mathbf{J}_0\}$ and for any $m, n \in \{0, \ldots, k_s - 1\}$,

$$\Big\| \sum_{j \in \mathrm{J}_0} \mathrm{P}_{j+s(m,n)}^{\top}(\boldsymbol{y}_j) \Big\|^2 = \sum_{j \in \mathrm{J}_0} ||\boldsymbol{y}_j||^2. \tag{4.26}$$

We now establish the first part of Proposition 4.14: if $\mathrm{D}_{\text{patch}}$ is contractive, then D must be contractive. Let $\beta \in [0, 1)$ be the contraction factor for $\mathrm{D}_{\text{patch}}$, i.e.,

$$\forall\, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^p : \quad \Big\| \mathrm{D}_{\text{patch}}(\boldsymbol{x}) - \mathrm{D}_{\text{patch}}(\boldsymbol{y}) \Big\| \leqslant \beta \, ||\boldsymbol{x} - \boldsymbol{y}|| \,. \tag{4.27}$$

By (4.23), D is average of of the operators $\{\mathrm{D}_{mn}\}$. Thus, by Lemma 4.5, it suffices to show that each D_{mn} is contractive. Indeed, let $\mathbf{X}_1, \mathbf{X}_2 \in \mathbb{R}^{q \times q}$. Then, from (4.25), (4.26) and (4.27), we can write

$$\begin{aligned}
&\big\| \mathrm{D}_{mn}(\mathbf{X}_1) - \mathrm{D}_{mn}(\mathbf{X}_2) \big\|^2 \\
&= \sum_{j \in \mathrm{J}_0} \big\| \mathrm{D}_{\text{patch}}(\mathrm{P}_{j+s(m,n)}(\mathbf{X}_1)) - \mathrm{D}_{\text{patch}}(\mathrm{P}_{j+s(m,n)}(\mathbf{X}_2)) \big\|^2 \\
&\leqslant \beta^2 \sum_{j \in \mathrm{J}_0} \big\| \mathrm{P}_{j+s(m,n)}(\mathbf{X}_1 - \mathbf{X}_2)) \big\|^2
\end{aligned}$$

$$= \beta^2 \left\| \mathbf{X}_1 - \mathbf{X}_2 \right\|^2.$$

Thus, each D_{mn} and hence D is contractive.

We next look at the second part of Proposition 4.14: if $\mathrm{D}_{\text{patch}}$ is θ-averaged, then D is θ-averaged. As before, we are done if we can show that each D_{mn} is θ-averaged by Lemma 4.5. Now, since $\mathrm{D}_{\text{patch}}$ is θ-averaged, there exists a nonexpansive operator $\mathrm{N}_{\text{patch}}$ such that

$$\mathrm{D}_{\text{patch}} = (1-\theta)\mathrm{I} + \theta \mathrm{N}_{\text{patch}}. \tag{4.28}$$

Substituting (4.28) in (4.23) and using (4.24), we can write

$$\mathrm{D}_{mn} = (1-\theta)\mathrm{I} + \theta \mathrm{G}_{mn},$$

where

$$\mathrm{G}_{mn} = \sum_{j \in \mathrm{J}_0} \mathrm{P}_{j+s(m,n)}^{\top} \mathrm{N}_{\text{patch}} \mathrm{P}_{j+s(m,n)}. \tag{4.29}$$

Then, using (4.25) and (4.26) once more and the nonexpansivity of $\mathrm{N}_{\text{patch}}$, we have for all $\mathbf{X}_1, \mathbf{X}_2 \in \mathbb{R}^{q \times q}$,

$$\left\| \mathrm{G}_{mn}(\mathbf{X}_1) - \mathrm{G}_{mn}(\mathbf{X}_2) \right\|^2 \leqslant \|\mathbf{X}_1 - \mathbf{X}_2\|^2.$$

Thus, we have shown that G_{mn} is nonexpansive and that D_{mn} is θ-averaged. This completes the proof of Proposition 4.14.

Conclusion

In this thesis, we focus on the convergence aspect of two iterative algorithms—Plug-and-Play (PnP) and Regularization by Denoising (RED) — a powerful denoiser is used for image regularization. More precisely, we came up with novel theoretical results to understand the convergence of PnP with linear denoisers and the convergence of PnP and RED with learning-based denoisers. Our results helped design algorithms that can match the regularization capabilities of state-of-the-art learning-based regularizers and come with provable convergence guarantees. The main contributions of our thesis are as follows.

- We showed that both iterate and objective convergence of PnP-ISTA and PnP-ADMM can be guaranteed for linear denoisers which are diagonalizable with eigenvalues in $[0, 1]$. In particular, we need to work with a denoiser-specific inner product (and associated gradient and proximal operators). Our results subsume existing convergence results for symmetric linear denoisers. Importantly, our analysis holds for non-symmetric kernel filters like nonlocal means which are known to possess good regularization capabilities. In fact, we demonstrated this for model-based superresolution and despeckling.

- We showed for the first time that solving a linear system is enough to obtain near to state-of-the-art regularization for linear inverse problems. This involves showing that PnP regularization using kernel denoisers like nonlocal means amounts to solving the classical regularization problem of minimizing $f + g$, where f is the loss term and g is a convex quadratic regularizer. We went on to prove that for linear inverse problems with quadratic f, the first-order optimality condition for this problem can be reduced to a linear system, which is solvable and admits a unique solution for deblurring, superresolution, and inpainting. Instead of performing PnP iterations, we proposed to directly solve this linear system. Indeed, using efficient Krylov solvers, we can solve this linear system at a superlinear rate which is a big jump from the sublinear convergence guarantee of PnP algorithms. We validated the speedup in practice using deblurring, superresolution, and inpainting experiments. In terms of reconstruction quality, we were able to get close to deep learning methods.

- We trained averaged (contractive) patch denoisers by unfolding ISTA and ADMM

applied to wavelet denoising and showed how they can be extended for image denoising using patch aggregation while preserving the averaged (contractive) property. Moreover, we showed that their regularization capacity for PnP and RED could be brought to par with BM3D and DnCNN by adding sufficiently many layers. Unlike existing CNN denoisers, we can guarantee convergence of PnP and RED using the proposed denoisers and this was verified numerically. To the best of our knowledge, this is the first work to exploit the properties of proximal operators in unfolded algorithms to come up with averaged and contractive denoisers.

We conclude by listing some possible extensions of the work carried out in this thesis.

- In Chapter 4, the result of our trained denoiser being contractive holds for any convex regularizer (not restricted to wavelet regularizer). Thus an important question is if we can go beyond classical wavelet denoising and come up with unfolded averaged (contractive) denoisers with superior denoising and regularization capacity.
- An interesting question arising from our analysis of PnP-ISTA and PnP-ADMM is whether we can establish the convergence of their accelerated variants FISTA [40] and A-ADMM [177]. The challenge lies in reducing these accelerated algorithms to a dynamic system of the form $\boldsymbol{x}_{k+1} = \mathrm{T}(\boldsymbol{x})$ for some contractive or averaged operator T.
- We have seen that nonlocal means provide impressive regularization capabilities for reconstructing high-dimensional images [178], which comes up in image fusion applications like HS-MS fusion and pansharpening [179]. Can we obtain state-of-the-art fusion performance by using Algorithm 1 in Chapter 3?
- In Chapter 3, we need to understand the relation between the number of PnP iterations to generate the guide image and the final reconstruction performance.
- We need to come up with a fast algorithm to generate the guide image. This will in effect reduce the computation time taken by the proposed method in Chapter 3.
- A theoretical framework, to extend our convergence analysis to derive performance bounds for proposed iterative frameworks, needs to be developed.
- We need to understand the scope of unfolded networks and our convergence results in other inverse problems in either image processing or other domains.
- Nonlocal means is preferred as a denoiser not only because of its denoising capabilities but also due to the existence of fast approximation algorithms for nonlocal means (including our works [179, 180]). An open problem is to check if these approximations satisfy the hypothesis required for the convergence of PnP and RED algorithms.

Bibliography

[1] K. Zhang, W. Ren, W. Luo, W.-S. Lai, B. Stenger, M.-H. Yang, and H. Li, "Deep image deblurring: A survey," *arXiv preprint arXiv:2201.10700*, 2022.

[2] J. Jam, C. Kendrick, K. Walker, V. Drouard, J. G.-S. Hsu, and M. H. Yap, "A comprehensive review of past and present image inpainting methods," *Comput. Vis. Image Underst.*, vol. 203, p. 103147, 2021.

[3] H. Boche, R. Calderbank, G. Kutyniok, and J. Vybíral, "A survey of compressed sensing," in *Compressed sensing and its applications*. Springer, 2015, pp. 1–39.

[4] K. Li, S. Yang, R. Dong, X. Wang, and J. Huang, "Survey of single image super-resolution reconstruction," *IET Image Process.*, vol. 14, no. 11, pp. 2273–2290, 2020.

[5] J. M. Bioucas-Dias and M. Figueiredo, "Multiplicative noise removal using variable splitting and constrained optimization," *IEEE Trans. Image Process.*, vol. 19, no. 7, pp. 1720–1730, 2010.

[6] A. Ribes and F. Schmitt, "Linear inverse problems in imaging," *IEEE Signal Process. Mag.*, vol. 25, no. 4, pp. 84–99, 2008.

[7] O. Scherzer, M. Grasmair, H. Grossauer, M. Haltmeier, and F. Lenzen, *Variational Methods in Imaging*. New York, NY, USA: Springer, 2009.

[8] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*. Dordrecht, Netherlands: Kluwer Academic Publishers, 1996.

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

[10] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly, "Compressed sensing MRI," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 72–82, 2008.

[11] G. T. Herman, *Fundamentals of computerized tomography: Image reconstruction from projections*. Springer Science & Business Media, 2009.

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

[13] A. Rond, R. Giryes, and M. Elad, "Poisson inverse problems by the plug-and-play scheme," *J. Vis. Commun. Image Represent.*, vol. 41, pp. 96–108, 2016.

[14] C. Fienup and J. Dainty, "Phase retrieval and image reconstruction for astronomy," *Image recovery: Theory and application*, vol. 231, p. 275, 1987.

[15] A. K. Jain, *Fundamentals of digital image processing*. Prentice-Hall, Inc., 1989.

[16] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 29, no. 6, pp. 1153–1160, 1981.

[17] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera, "Filling-in by joint interpolation of vector fields and gray levels," *IEEE Trans. Image Process.*, vol. 10, no. 8, pp. 1200–1211, 2001.

[18] L. Landweber, "An iteration formula for fredholm integral equations of the first kind," *Amer. J. Math.*, vol. 73, no. 3, pp. 615–624, 1951.

[19] W. Dong, L. Zhang, G. Shi, and X. Wu, "Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization," *IEEE Trans. Image Process.*, vol. 20, no. 7, pp. 1838–1857, 2011.

[20] G. Jagatap and C. Hegde, "Algorithmic guarantees for inverse imaging with untrained network priors," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 14 832–14 842, 2019.

[21] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D*, vol. 60, no. 1–4, pp. 259–268, 1992.

[22] A. Chambolle, R. A. De Vore, N.-Y. Lee, and B. J. Lucier, "Nonlinear wavelet image processing: variational problems, compression, and noise removal through wavelet shrinkage," *IEEE Trans. Image Process.*, vol. 7, no. 3, pp. 319–335, 1998.

[23] A. N. Tikhonov, "On the solution of ill-posed problems and the method of regularization," *Doklady Akademii Nauk*, vol. 151, no. 3, pp. 501–504, 1963.

[24] T. Chan, A. Marquina, and P. Mulet, "High-order total variation-based image restoration," *SIAM J. Sci. Comput.*, vol. 22, no. 2, pp. 503–516, 2000.

[25] K. Bredies, K. Kunisch, and T. Pock, "Total generalized variation," *SIAM J. Imaging Sci.*, vol. 3, no. 3, pp. 492–526, 2010.

[26] J. Myrheim and H. Rue, "New algorithms for maximum entropy image restoration," *CVGIP: Graph. Models Image Process.*, vol. 54, no. 3, pp. 223–238, 1992.

[27] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J Fourier Anal Appl*, vol. 14, no. 5, pp. 877–905, 2008.

[28] W. Dong, X. Li, L. Zhang, and G. Shi, "Sparsity-based image denoising via dictionary learning and structural clustering," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, pp. 457–464, 2011.

[29] J. Sun and M. F. Tappen, "Learning non-local range markov random field for image restoration," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, pp. 2745–2752, 2011.

[30] S. Lefkimmiatis, J. P. Ward, and M. Unser, "Hessian schatten-norm regularization for linear inverse problems," *IEEE Trans. Image Process.*, vol. 22, no. 5, pp. 1873–1888, 2013.

[31] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," in *Proc. IEEE Intl. Conf. Comp. Vis.*, 2011, pp. 479–486.

[32] A. Elmoataz, O. Lezoray, and S. Bougleux, "Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing," *IEEE Trans. Image Process.*, vol. 17, no. 7, pp. 1047–1060, 2008.

[33] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, 2006.

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

[35] L. Zhang and W. Zuo, "Image restoration: From sparse and low-rank priors to deep priors [lecture notes]," *IEEE Signal Process. Mag.*, vol. 34, no. 5, pp. 172–179, 2017.

[36] D. Bertsekas and A. Nedic, *Convex Analysis and Optimization*. Athena Scientific, 2003.

[37] D. P. Bertsekas, "Nonlinear programming," *Journal of the Operational Research Society*, vol. 48, no. 3, pp. 334–334, 1997.

[38] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[39] A. Beck, *First-order methods in optimization*. SIAM, 2017.

[40] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imaging Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[41] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[42] J. Eckstein and D. P. Bertsekas, "On the Douglas—Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Math. Program.*, vol. 55, no. 1-3, pp. 293–318, 1992.

[43] D. Geman and G. Reynolds, "Constrained restoration and the recovery of discontinuities," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 3, pp. 367–383, 1992.

[44] G. Ongie, A. Jalal, C. A. Metzler, R. G. Baraniuk, A. G. Dimakis, and R. Willett, "Deep learning techniques for inverse problems in imaging," *IEEE J. Sel. Areas Inf. Theory*, vol. 1, no. 1, pp. 39–56, 2020.

[45] W. Dong, P. Wang, W. Yin, G. Shi, F. Wu, and X. Lu, "Denoising prior driven deep neural network for image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 10, pp. 2305–2318, 2018.

[46] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, "Deep convolutional neural network for inverse problems in imaging," *IEEE Trans. Image Process.*, vol. 26, no. 9, pp. 4509–4522, 2017.

[47] M. T. McCann, K. H. Jin, and M. Unser, "Convolutional neural networks for inverse problems in imaging: A review," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 85–95, 2017.

[48] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," *Proc. Intl. Conf. Mach. Learn.*, pp. 399–406, 2010.

[49] D. Gilton, G. Ongie, and R. Willett, "Neumann networks for linear inverse problems in imaging," *IEEE Trans. Comput. Imag.*, vol. 6, pp. 328–343, 2019.

[50] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, pp. 9446–9454, 2018.

[51] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 295–307, 2015.

[52] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, pp. 1646–1654, 2016.

[53] L. Xu, J. S. Ren, C. Liu, and J. Jia, "Deep convolutional neural network for image deconvolution," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014.

[54] W. Shi, F. Jiang, S. Liu, and D. Zhao, "Image compressed sensing using convolutional neural network," *IEEE Trans. Image Process.*, vol. 29, pp. 375–388, 2019.

[55] J. Park, D. Hwang, K. Y. Kim, S. K. Kang, Y. K. Kim, and J. S. Lee, "Computed tomography super-resolution using deep convolutional neural network," *Phys. Med. Biol.*, vol. 63, no. 14, p. 145011, 2018.

[56] P. Wang, H. Zhang, and V. M. Patel, "Sar image despeckling using a convolutional neural network," *IEEE Signal Process. Lett.*, vol. 24, no. 12, pp. 1763–1767, 2017.

[57] T. Remez, O. Litany, R. Giryes, and A. M. Bronstein, "Class-aware fully convolutional gaussian and poisson denoising," *IEEE Trans. Image Process.*, vol. 27, no. 11, pp. 5707–5722, 2018.

[58] J. Zhang and B. Ghanem, "ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing," *Proc. IEEE Intl. Conf. Comp. Vis.*, pp. 1828–1837, 2018.

[59] D. Liang, J. Cheng, Z. Ke, and L. Ying, "Deep magnetic resonance image reconstruction: Inverse problems meet neural networks," *IEEE Signal Process. Mag.*, vol. 37, no. 1, pp. 141–151, 2020.

[60] K. Zhang, L. V. Gool, and R. Timofte, "Deep unfolding network for image super-resolution," *Proc. IEEE Intl. Conf. Comp. Vis.*, pp. 3217–3226, 2020.

[61] J. Sun, H. Li, Z. Xu *et al.*, "Deep admm-net for compressive sensing mri," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016.

[62] R. Aljadaany, D. K. Pal, and M. Savvides, "Douglas-rachford networks: Learning both the image prior and data fidelity terms for blind image deconvolution," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, pp. 10 235–10 244, 2019.

[63] E. Ahishakiye, M. Bastiaan Van Gijzen, J. Tumwiine, R. Wario, and J. Obungoloch, "A survey on deep learning in medical image reconstruction," *Intelligent Medicine*, vol. 1, no. 03, pp. 118–127, 2021.

[64] Z. YiNan and A. MingQiang, "Deep learning-and transfer learning-based super resolution reconstruction from single medical image," *J. Healthc. Eng.*, vol. 2017, 2017.

[65] L. P. Yaroslavsky, *Digital Picture Processing*. Berlin, Germany: Springer-Verlag, 1985.

[66] J.-S. Lee, "Digital image smoothing and the sigma filter," *Comp. Vis. Graph. Image Process.*, vol. 24, no. 2, pp. 255–269, 1983.

[67] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," *Proc. IEEE Intl. Conf. Comp. Vis.*, pp. 839–846, 1998.

[68] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, vol. 2, pp. 60–65, 2005.

[69] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Trans. Image Process.*, vol. 16, no. 2, pp. 349–366, 2007.

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

[71] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1256–1272, 2016.

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

[73] K. Zhang, W. Zuo, and L. Zhang, "FFDNet: Toward a fast and flexible solution for CNN-based image denoising," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4608–4622, 2018.

[74] P. Liu, H. Zhang, K. Zhang, L. Lin, and W. Zuo, "Multi-level wavelet-CNN for image restoration," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit. Workshops*, pp. 773–782, 2018.

[75] D. Liu, B. Wen, Y. Fan, C. C. Loy, and T. S. Huang, "Non-local recurrent network for image restoration," *Proc. Adv. Neural Inf. Process. Syst.*, pp. 1673–1682, 2018.

[76] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, "Plug-and-play image restoration with deep denoiser prior," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021.

[77] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," *Proc. IEEE Global Conf. Signal and Information Process.*, pp. 945–948, 2013.

[78] E. Ryu, J. Liu, S. Wang, X. Chen, Z. Wang, and W. Yin, "Plug-and-play methods provably converge with properly trained denoisers," *Proc. Intl. Conf. Mach. Learn.*, vol. 97, pp. 5546–5557, 2019.

[79] S. Sreehari, S. V. Venkatakrishnan, B. Wohlberg, G. T. Buzzard, L. F. Drummy, J. P. Simmons, and C. A. Bouman, "Plug-and-play priors for bright field electron tomography and sparse interpolation," *IEEE Trans. Comput. Imag.*, vol. 2, no. 4, pp. 408–423, 2016.

[80] M. Borgerding, P. Schniter, and S. Rangan, "Amp-inspired deep networks for sparse linear inverse problems," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4293–4308, 2017.

[81] Y. Sun, B. Wohlberg, and U. S. Kamilov, "An online plug-and-play algorithm for regularized image reconstruction," *IEEE Trans. Comput. Imag.*, vol. 5, no. 3, pp. 395–408, 2019.

[82] Y. Sun, Z. Wu, X. Xu, B. Wohlberg, and U. S. Kamilov, "Scalable plug-and-play ADMM with convergence guarantees," *IEEE Trans. Comput. Imag.*, vol. 7, pp. 849–863, 2021.

[83] J. Rick Chang, C.-L. Li, B. Poczos, B. V. K. Vijaya Kumar, and A. C. Sankaranarayanan, "One network to solve them all–solving linear inverse problems using deep projection models," *Proc. IEEE Intl. Conf. Comp. Vis.*, pp. 5888–5897, 2017.

[84] K. Zhang, W. Zuo, and L. Zhang, "Deep plug-and-play super-resolution for arbitrary blur kernels," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, pp. 1671–1681, 2019.

[85] U. S. Kamilov, H. Mansour, and B. Wohlberg, "A plug-and-play priors approach for solving nonlinear imaging inverse problems," *IEEE Signal Process. Lett.*, vol. 24, no. 12, pp. 1872–1876, 2017.

[86] S. Ono, "Primal-dual plug-and-play image restoration," *IEEE Signal Process. Lett.*, vol. 24, no. 8, pp. 1108–1112, 2017.

[87] T. Tirer and R. Giryes, "Image restoration by iterative denoising and backward projections," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1220–1234, 2019.

[88] A. M. Teodoro, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "A convergent image fusion algorithm using scene-adapted Gaussian-mixture-based denoising," *IEEE Trans. Image Process.*, vol. 28, no. 1, pp. 451–463, 2019.

[89] G. Song, Y. Sun, J. Liu, Z. Wang, and U. S. Kamilov, "A new recurrent plug-and-play prior based on the multiple self-similarity network," *IEEE Signal Process. Lett.*, vol. 27, pp. 451–455, 2020.

[90] R. Ahmad, C. A. Bouman, G. T. Buzzard, S. Chan, S. Liu, E. T. Reehorst, and P. Schniter, "Plug-and-play methods for magnetic resonance imaging: Using denoisers for image recovery," *IEEE Signal Process. Mag.*, vol. 37, no. 1, pp. 105–116, 2020.

[91] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (RED)," *SIAM J. Imaging Sci.*, vol. 10, no. 4, pp. 1804–1844, 2017.

[92] E. T. Reehorst and P. Schniter, "Regularization by denoising: Clarifications and new interpretations," *IEEE Trans. Comput. Imag.*, vol. 5, no. 1, pp. 52–67, 2018.

[93] C. Metzler, P. Schniter, A. Veeraraghavan *et al.*, "prDeep: robust phase retrieval with a flexible deep network," *International Conference on Machine Learning*, pp. 3501–3510, 2018.

[94] Z. Wu, Y. Sun, J. Liu, and U. Kamilov, "Online regularization by denoising with applications to phase retrieval," *Proc. IEEE Intl. Conf. Comp. Vis. Wkshp.*, pp. 0–0, 2019.

[95] Z. Wu, Y. Sun, A. Matlock, J. Liu, L. Tian, and U. S. Kamilov, "Simba: Scalable inversion in optical tomography using deep denoising priors," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 6, pp. 1163–1175, 2020.

[96] G. Mataev, P. Milanfar, and M. Elad, "DeepRED: Deep image prior powered by RED," *Proc. IEEE Intl. Conf. Comp. Vis. Wksh.*, 2019.

[97] Y. Sun, J. Liu, and U. S. Kamilov, "Block coordinate regularization by denoising," *Proc. Adv. Neural Inf. Process. Syst.*, pp. 380–390, 2019.

[98] Y. Hu, J. Liu, X. Xu, and U. S. Kamilov, "Monotonically convergent regularization by denoising," *arXiv preprint arXiv:2202.04961*, 2022.

[99] J. Hertrich, S. Neumayer, and G. Steidl, "Convolutional proximal neural networks and plug-and-play algorithms," *Linear Algebra Appl.*, vol. 631, pp. 203–234, 2021.

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

[101] T. Meinhardt, M. Moller, C. Hazirbas, and D. Cremers, "Learning proximal operators: Using denoising networks for regularizing inverse imaging problems," *Proc. IEEE Intl. Conf. Comp. Vis.*, pp. 1781–1790, 2017.

[102] S. H. Chan, "Performance analysis of plug-and-play ADMM: A graph signal processing perspective," *IEEE Trans. Comput. Imag.*, vol. 5, no. 2, pp. 274–286, 2019.

[103] R. G. Gavaskar and K. N. Chaudhury, "Plug-and-play ISTA converges with kernel denoisers," *IEEE Signal Process. Lett.*, vol. 27, pp. 610–614, 2020.

[104] X. Yuan, Y. Liu, J. Suo, and Q. Dai, "Plug-and-play algorithms for large-scale snapshot compressive imaging," *arXiv preprint arXiv:2003.13654*, 2020.

[105] X. Xu, Y. Sun, J. Liu, B. Wohlberg, and U. S. Kamilov, "Provable convergence of plug-and-play priors with MMSE denoisers," *IEEE Signal Process. Lett.*, vol. 27, pp. 1280–1284, 2020.

[106] J. Liu, S. Asif, B. Wohlberg, and U. Kamilov, "Recovery analysis for plug-and-play priors using the restricted eigenvalue condition," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021.

[107] A. H. Al-Shabili, X. Xu, I. Selesnick, and U. S. Kamilov, "Bregman plug-and-play priors," *arXiv preprint arXiv:2202.02388*, 2022.

[108] S. Hurault, A. Leclaire, and N. Papadakis, "Gradient step denoiser for convergent plug-and-play," *Proc. Int. Conf. Learn. Represent.*, 2022.

[109] R. Cohen, Y. Blau, D. Freedman, and E. Rivlin, "It has potential: Gradient-driven denoisers for convergent solutions to inverse problems," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021.

[110] S. Hurault, A. Leclaire, and N. Papadakis, "Proximal denoiser for convergent plug-and-play optimization with nonconvex regularization," *arXiv preprint arXiv:2201.13256*, 2022.

[111] A. Raj, Y. Li, and Y. Bresler, "GAN-based projector for faster recovery with convergence guarantees in linear inverse problems," *Proc. IEEE Intl. Conf. Comp. Vis.*, pp. 5602–5611, 2019.

[112] Y. Sun, Z. Wu, B. Wohlberg, and U. S. Kamilov, "Scalable plug-and-play admm with convergence guarantees," *arXiv preprint arXiv:2006.03224*, vol. 7, pp. 849–863, 2020.

[113] R. Cohen, M. Elad, and P. Milanfar, "Regularization by denoising via fixed-point projection (RED-PRO)," *arXiv preprint arXiv:2008.00226*, 2020.

[114] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, 2nd ed. New York, NY, USA: Springer, 2017.

[115] R. Cohen, M. Elad, and P. Milanfar, "Regularization by denoising via fixed-point projection (red-pro)," *SIAM J. Imaging Sci.*, vol. 14, no. 3, pp. 1374–1406, 2021.

[116] A. Virmaux and K. Scaman, "Lipschitz regularity of deep neural networks: Analysis and efficient estimation," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, pp. 3835–3844, 2018.

[117] H. Sedghi, V. Gupta, and P. M. Long, "The singular values of convolutional layers," *Proc. Intl. Conf. Learn. Represent.*, 2019.

[118] M. Terris, A. Repetti, J.-C. Pesquet, and Y. Wiaux, "Building firmly nonexpansive convolutional neural networks," *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, pp. 8658–8662, 2020.

[119] J.-C. Pesquet, A. Repetti, M. Terris, and Y. Wiaux, "Learning maximally monotone operators for image recovery," *SIAM J. Imaging Sci.*, vol. 14, no. 3, pp. 1206–1237, 2021.

[120] A. K. Fletcher, P. Pandit, S. Rangan, S. Sarkar, and P. Schniter, "Plug-in estimation in high-dimensional linear inverse problems: A rigorous analysis," *Proc. Adv. Neural Inf. Process. Syst.*, pp. 7440–7449, 2018.

[121] W. Rudin, *Principles of Mathematical Analysis*, 3rd ed. New York, NY, USA: McGraw-Hill, 1976.

[122] C.-A. Deledalle, L. Denis, S. Tabti, and F. Tupin, "MuLoG, or how to apply Gaussian denoisers to multi-channel SAR speckle reduction?" *IEEE Trans. Image Process.*, vol. 26, no. 9, pp. 4389–4403, 2017.

[123] W. Dong, L. Zhang, G. Shi, and X. Li, "Nonlocally centralized sparse representation for image restoration," *IEEE Trans. Image Process.*, vol. 22, no. 4, pp. 1620–1630, 2012.

[124] I. Ram, M. Elad, and I. Cohen, "Image processing using smooth ordering of its patches," *IEEE Trans. Image Process.*, vol. 22, no. 7, pp. 2764–2774, 2013.

[125] M. K. Ng, P. Weiss, and X. Yuan, "Solving constrained total-variation image restoration and reconstruction problems via alternating direction methods," *SIAM J. Sci. Comput.*, vol. 32, no. 5, pp. 2710–2736, 2010.

[126] N. Zhao, Q. Wei, A. Basarab, N. Dobigeon, D. Kouamé, and J.-Y. Tourneret, "Fast single image super-resolution using a new analytical solution for $\ell_2 - \ell_2$ problems," *IEEE Trans. Image Process.*, vol. 25, no. 8, pp. 3683–3697, 2016.

[127] P. Milanfar, "A tour of modern image filtering: New insights and methods, both practical and theoretical," *IEEE Signal Process. Mag.*, vol. 30, no. 1, pp. 106–128, 2013.

[128] F. Heide, M. Steinberger, Y.-T. Tsai, M. Rouf, D. Pajak, D. Reddy, O. Gallo, J. Liu, W. Heidrich, K. Egiazarian, J. Kautz, and K. Pulli, "Flexisp: A flexible camera image processing framework," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 1–13, 2014.

[129] S. Sreehari, S. Venkatakrishnan, K. L. Bouman, J. P. Simmons, L. F. Drummy, and C. A. Bouman, "Multi-resolution data fusion for super-resolution electron microscopy," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit. Wksh.*, pp. 88–96, 2017.

[130] V. S. Unni, S. Ghosh, and K. N. Chaudhury, "Linearized ADMM and fast nonlocal denoising for efficient plug-and-play restoration," *Proc. IEEE Global Conf. Signal and Information Process.*, pp. 11–15, 2018, [Online]. https://arxiv.org/abs/1901.06110/.

[131] P. Nair, V. S. Unni, and K. N. Chaudhury, "Hyperspectral image fusion using fast high-dimensional denoising," *Proc. IEEE Intl. Conf. Image Process.*, pp. 3123–3127, 2019.

[132] A. Singer, Y. Shkolnisky, and B. Nadler, "Diffusion interpretation of nonlocal neighborhood filters for signal denoising," *SIAM J. Imaging Sci.*, vol. 2, no. 1, pp. 118–139, 2009.

[133] J. P. Morel, A. Buades, and T. Coll, "Local smoothing neighborhood filters," in *Handbook of Mathematical Methods in Imaging*, 2nd ed. New York, NY, USA: Springer, 2015.

[134] P. Milanfar, "Symmetrizing smoothing filters," *SIAM J. Imaging Sci.*, vol. 6, no. 1, pp. 263–284, 2013.

[135] "Matlab code," matlab and Python code for scaled PnP algorithms. [Online]. Available: https://github.com/pravin1390/ScaledPnP. [Online]. Available: https://github.com/pravin1390/FastHDNystrom

[136] P. L. Combettes and J.-C. Pesquet, "Deep neural network structures solving variational inequalities," *Set-Valued Variational Anal.*, pp. 1–28, 2020.

[137] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *Proc. Intl. Conf. Learn. Represent.*, 2018.

[138] H. Gouk, E. Frank, B. Pfahringer, and M. Cree, "Regularisation of neural networks by enforcing Lipschitz continuity," *arXiv preprint arXiv:1804.04368*, 2018.

[139] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," *Proc. IEEE Intl. Conf. Comp. Vis.*, vol. 2, pp. 416–423, 2001.

[140] S. Boyd and L. Vandenberghe, *Convex Optimization*. CU Press, 2004.

[141] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[142] G. H. Golub and C. F. Van Loan, *Matrix Computations*. JHU Press, 2012, vol. 3.

[143] J. Liu, X. Chen, Z. Wang, and W. Yin, "ALISTA: Analytic weights are as good as learned weights in LISTA," *Proc. Intl. Conf. Learn. Represent.*, 2019.

[144] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

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

[146] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution as sparse representation of raw image patches," in *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, 2008, pp. 1–8.

[147] H. He and W.-C. Siu, "Single image super-resolution using Gaussian process regression," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, pp. 449–456, 2011.

[148] T. Peleg and M. Elad, "A statistical prediction model based on sparse representations for single image super-resolution," *IEEE Trans. Image Process.*, vol. 23, no. 6, pp. 2569–2582, 2014.

[149] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, pp. 5197–5206, 2015.

[150] A. Chambolle, "An algorithm for total variation minimization and applications," *J. Math. Imag. Vis.*, vol. 20, no. 1-2, pp. 89–97, 2004.

[151] S. Parrilli, M. Poderico, C. V. Angelino, and L. Verdoliva, "A nonlocal SAR image denoising algorithm based on LLMMSE wavelet shrinkage," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 2, pp. 606–616, 2011.

[152] C. J. Schuler, H. C. Burger, S. Harmeling, and B. Scholkopf, "A machine learning approach for non-blind image deconvolution," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, pp. 1067–1074, 2013.

[153] A. Levin, R. Fergus, F. Durand, and W. T. Freeman, "Image and depth from a conventional camera with a coded aperture," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 70–es, 2007.

[154] D. Krishnan and R. Fergus, "Fast image deconvolution using hyper-Laplacian priors," *Proc. Adv. Neural Inf. Process. Syst.*, pp. 1033–1041, 2009.

[155] A. Danielyan, V. Katkovnik, and K. Egiazarian, "BM3D frames and variational image deblurring," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1715–1728, 2011.

[156] S. Roth and M. J. Black, "Fields of experts: A framework for learning image priors," *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, vol. 2, pp. 860–867, 2005.

[157] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Boston, MA, USA: Springer, 2004.

[158] P. L. Combettes, "Monotone operator theory in convex optimization," *Math. Program.*, vol. 170, no. 1, pp. 177–206, 2018.

[159] E. W. Cheney and W. A. Light, *A Course in Approximation Theory*. Providence, RI, USA: American Mathematical Society, 2009.

[160] L. N. Trefethen and D. Bau III, *Numerical Linear Algebra*. SIAM, 1997, vol. 50.

[161] J. R. Shewchuk *et al.*, *An introduction to the conjugate gradient method without the agonizing pain*. Carnegie-Mellon University. Department of Computer Science, 1994.

[162] Y. Saad and M. H. Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Statis. Comput.*, vol. 7, no. 3, pp. 856–869, 1986.

[163] A. H. Baker, E. R. Jessup, and T. Manteuffel, "A technique for accelerating the convergence of restarted GMRES," *SIAM J. Matrix Anal. Appl.*, vol. 26, no. 4, pp. 962–984, 2005.

[164] J. E. Hicken and D. W. Zingg, "A simplified and flexible variant of GCROT for solving nonsymmetric linear systems," *SIAM J. Sci. Comput.*, vol. 32, no. 3, pp. 1672–1694, 2010.

[165] C. G. Broyden, "A class of methods for solving nonlinear simultaneous equations," *Math. Comput.*, vol. 19, no. 92, pp. 577–593, 1965.

[166] J. J. Moré and J. A. Trangenstein, "On the global convergence of Broyden's method," *Math. Comput.*, vol. 30, no. 135, pp. 523–540, 1976.

[167] B. He and X. Yuan, "On the $\mathcal{O}(1/n)$ convergence rate of the Douglas–Rachford alternating direction method," *SIAM J. Numer. Anal.*, vol. 50, no. 2, pp. 700–709, 2012.

[168] R. G. Gavaskar, C. D. Athalye, and K. N. Chaudhury, "On plug-and-play regularization using linear denoisers," *IEEE Trans. Image Process.*, vol. 30, pp. 4802–4813, 2021.

[169] V. S. Unni, P. Nair, and K. N. Chaudhury, "Plug-and-play registration and fusion," *IEEE Intl. Conf. Image Process.*, pp. 2546–2550, 2020.

[170] E. H. Moore, "On the reciprocal of the general algebraic matrix," *Bull. Am. Math. Soc.*, vol. 26, pp. 394–395, 1920.

[171] V. Simoncini and D. B. Szyld, "On the occurrence of superlinear convergence of exact and inexact krylov subspace methods," *SIAM Rev.*, vol. 47, no. 2, pp. 247–272, 2005.

[172] A. M. Teodoro, J. M. Bioucas-Dias, and M. A. Figueiredo, "A convergent image fusion algorithm using scene-adapted gaussian-mixture-based denoising," *IEEE Trans. Image Process.*, vol. 28, no. 1, pp. 451–463, 2019.

[173] ——, "Image restoration and reconstruction using targeted plug-and-play priors," *IEEE Trans. Comput. Imag.*, vol. 5, no. 4, pp. 675–686, 2019.

[174] S. Sreehari, S. V. Venkatakrishnan, L. Drummy, J. Simmons, and C. A. Bouman, "Rotationally-invariant non-local means for image denoising and tomography," *Proc. IEEE Intl. Conf. Image Process.*, pp. 542–546, 2015.

[175] E. K. Ryu and S. Boyd, "Primer on monotone operator methods," *Appl. Comput. Math*, vol. 15, no. 1, pp. 3–43, 2016.

[176] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Understanding and evaluating blind deconvolution algorithms," in *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, 2009, pp. 1964–1971.

[177] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk, "Fast alternating direction optimization methods," *SIAM J. Imaging Sci.*, vol. 7, no. 3, pp. 1588–1623, 2014.

[178] R. Dian, S. Li, B. Sun, and A. Guo, "Recent advances and new guidelines on hyperspectral and multispectral image fusion," *Information Fusion*, vol. 69, pp. 40–51, 2021.

[179] P. Nair and K. N. Chaudhury, "Fast high-dimensional bilateral and nonlocal means filtering," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1470–1481, 2019.

[180] ——, "Fast high-dimensional kernel filtering," *IEEE Signal Process. Lett.*, vol. 26, no. 2, pp. 377–381, 2019.