Provably Convergent Plug-and-Play Quasi-Newton Methods

Tan, Hong Ye; Mukherjee, Subhadip; Tang, Junqi; Schönlieb, Carola-Bibiane

DOI:
10.1137/23M157185X

Provably Convergent Plug-and-Play Quasi-Newton Methods

Hong Ye Tan*, Subhadip Mukherjee*†, Junqi Tang*‡, and Carola-Bibiane Schönlieb*

Abstract. Plug-and-Play (PnP) methods are a class of efficient iterative methods that aim to combine data fidelity terms and deep denoisers using classical optimization algorithms, such as ISTA or ADMM, with applications in inverse problems and imaging. Provable PnP methods are a subclass of PnP methods with convergence guarantees, such as fixed point convergence or convergence to critical points of some energy function. Many existing provable PnP methods impose heavy restrictions on the denoiser or fidelity function, such as *nonexpansiveness* or *strict convexity*, respectively. In this work, we propose a novel algorithmic approach incorporating quasi-Newton steps into a provable PnP framework based on proximal denoisers, resulting in greatly accelerated convergence while retaining light assumptions on the denoiser. By characterizing the denoiser as the proximal operator of a weakly convex function, we show that the fixed points of the proposed quasi-Newton PnP algorithm are critical points of a weakly convex function. Numerical experiments on image deblurring and super-resolution demonstrate 2–8x faster convergence as compared to other provable PnP methods with similar reconstruction quality.

Key words. Plug-and-Play, inverse problems, quasi-Newton methods, image reconstruction

MSC codes. 49M15, 49J52, 65K15

1. Introduction.

Many image restoration problems can be formulated as reconstructing data $x \in \mathbb{R}^n$ from a noisy measurement $y = Ax + \varepsilon \in \mathbb{R}^m$, where A is a linear forward operator, and ε is some measurement noise. One common way to solve this is the variational formulation

$$\arg\min_{x \in \mathbb{R}^n} \varphi(x) = f(x) + g(x), \tag{1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is typically a continuously differentiable data fidelity term, and $g : \mathbb{R}^n \to \overline{\mathbb{R}}$ is a regularization term that controls the prior. In many cases, the fidelity term incorporates a forward operator $A : \mathbb{R}^n \to \mathbb{R}^m$, which may correspond to physical operators such as blurring operators or Radon transforms [28]. For a noisy measurement $y = Ax + \varepsilon$ with additive white noise $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$, the fidelity term takes the form of the negative log likelihood $f(x) = \|Ax - y\|^2/(2\sigma^2)$. For many physical forward operators, such as blurring or down-sampling, the optimization problem $\min_x f(x)$ is ill-posed, thus a regularization term is needed [36]. Classical examples for regularization include using Fourier spectra (spectral regularization) or total variation (TV) regularization on natural images [62, 63], whereas recent works aim to learn a neural network regularizer [44, 49].

Fully data-driven approaches have been shown to outperform explicitly defined regularizers [77, 76, 49]. However, the outputs of these learned schemes often do not correspond to

*Department of Applied Mathematics and Theoretical Physics, University of Cambridge, UK (hyt35@cam.ac.uk, sm2467@cam.ac.uk, cbs31@cam.ac.uk).

†Department of Computer Science, University of Bath, UK. Department of Electronics and Electrical Communication Engineering, Indian Institute of Technology, Kharagpur, India (smukherjee@ece.iitkgp.ac.in).

‡School of Mathematics, University of Birmingham, UK (j.tang.2@bham.ac.uk).

closed-form minimization problems of the form (1.1). This is particularly limiting in sensitive applications such as medical imaging, where interpretability is necessary [73, 72]. Recent lines of work consider combining iterative algorithms with generic denoisers, with notable examples including regularization by denoising (RED) [17, 58], consensus equilibrium [12], and deep mean-shift priors [2]. In this work, we will focus on the line of Plug-and-Play (PnP) methods, which arise from replacing proximal steps with denoisers. Under certain conditions on the fidelity and denoisers as detailed in Section 1.2, fixed point convergence of certain PnP methods can be established, characterized by critical points of a corresponding functional.

The PnP framework of replacing the regularization proximal step with a denoiser is flexible in the choice of denoiser. In particular, it allows for the use of both classical denoisers such as NLM or BM3D [11, 18], as well as data-driven denoisers [78, 77, 64]. This allows for extending the use of Gaussian denoisers to other image reconstruction tasks, such as super-resolution or image deblocking. Recently, PnP methods based on the half-quadratic splitting were able to achieve state-of-the-art performance for image reconstruction using a variable-strength Gaussian denoiser called DRUNet [78]. Named the deep Plug-and-Play image restoration (DPIR) method, DPIR outperforms or is competitive with fully learned methods for applications such as image deblurring, super-resolution, and demosaicing while using only a single denoiser prior [77]. This work demonstrates the flexibility of PnP, using one prior for multiple reconstruction tasks.

While PnP methods can be used to achieve excellent performance, empirical convergence does not equate to traditional notions of convergence. Indeed, while DPIR is able to achieve state-of-the-art results in as few as eight PnP iterations, there are no associated theoretical results. Moreover, DPIR can diverge when more PnP iterations are applied [32]. This can be empirically alleviated using various stopping criteria, but this raises an additional issue for defining a notion of "best reconstruction". In this work, we sidestep this by considering provable PnP methods. We use the term "provable PnP" to refer to PnP methods equipped with some notion of convergence, such as fixed-point convergence, or the stronger notion of convergence to critical points of a function.

Various approaches for accelerating PnP methods have been proposed, including using classical accelerated optimization algorithms, block-coordinate methods, parallelization, and dimensionality reduction [38, 23, 71, 37, 68]. In the context of convergence to fixed points of a functional, theoretical results for PnP based on accelerated classical methods such as FISTA have not arisen in the literature. This work proposes to extend the work on provable PnP methods by introducing a quasi-Newton step to accelerate convergence, while retaining a corresponding closed-form minimization problem with relatively weak constraints.

1.1. Definitions and Notations. We begin with some definitions and notation. Let $\overline{\mathbb{R}} = \mathbb{R} \cup \{+\infty\}$ be the extended real line. Recall that a function $g : \mathbb{R}^n \to \overline{\mathbb{R}}$ is *proper* if the effective domain $\operatorname{dom} g = \{x \in \mathbb{R}^n \mid g(x) < +\infty\}$ is nonempty, and *closed* (or *lower-semicontinuous*) if for every sequence $x^k \to x$ in $\mathbb{R}^n$, we have $g(x) \leq \liminf_k g(x^k)$.

Definition 1.1. *For a scalar $\gamma > 0$ and a proper closed convex function $g : \mathbb{R}^n \to \overline{\mathbb{R}}$, the* proximal map *is*

$$\operatorname{prox}_{\gamma g}(x) = \operatorname*{arg\,min}_{u \in \mathbb{R}^n} \left\{ g(u) + \frac{1}{2\gamma}\|u - x\|^2 \right\}. \tag{1.2}$$

The Moreau envelope *is the value function of the proximal map, defined as*

$$g^\gamma(x) = \min_{u\in\mathbb{R}^n} \left\{ g(u) + \frac{1}{2\gamma}\|u - x\|^2 \right\}. \tag{1.3}$$

Properties of the Moreau envelope and proximal operators are well documented in classical literature [59, 7, 48, 27]. In particular, for proper closed convex g, the proximal operator is single-valued and nonexpansive, and the envelope function g^γ is convex and $\mathcal{C}^1$ with derivative

$$\nabla g^\gamma(x) = \gamma^{-1}(x - \mathrm{prox}_{\gamma g}(x)).$$

1.2. Plug-and-Play Methods.

The Plug-and-Play (PnP) framework was first introduced by Venkatakrishnan et al. in 2013 for model-based image reconstruction [74]. PnP methods arise from composite convex optimization algorithms, wherein a prior regularization step is associated with a denoising step. The first composite optimization algorithm considered was Alternating Directions Method of Multipliers (ADMM), a classical proximal splitting algorithm used for minimizing composite functions. In the case of image reconstruction, a maximum likelihood estimation model can be decomposed into a composite problem. For a noisy measurement y and unknown data x, let $p(y|x)$ be the conditional likelihood, and $p(x)$ the prior of the unknown x. The maximum a-posteriori (MAP) estimate $\hat{x}$ is given as follows:

$$\begin{aligned}\hat{x} &= \arg\max_x \{p(y|x) + p(x)\}\\ &= \arg\min_x \{f(x;y) + g(x)\},\end{aligned}$$

where f is the likelihood/fidelity term, and g is the prior/regularization term. A classical example would be TV regularization for additive Gaussian noise, where the fidelity term is $f(x;y) = \|Ax-y\|_2^2/2\sigma^2$, and the prior term is $g(x) = \lambda\|\nabla x\|_1$ [63]. To solve the minimization problem for general convex f, g, proximal splitting algorithms such as ADMM consider alternating applications of the individual proximal operators $\mathrm{prox}_f, \mathrm{prox}_g$ or subgradients $\partial f, \partial g$. The key observation of PnP is that the prior regularization step can also be interpreted as a denoising operation [64].

More generally, the PnP framework can be applied to monotone operator splitting methods. Under light conditions, the composite convex optimization problem of minimizing $f + g$ can be reformulated as the monotone inclusion problem $0 \in \partial f(x) + \partial g(x)$ [59, 7]. For convex f and g, the operators ∂f and ∂g are monotone operators. Monotone operator splitting methods aim to solve the inclusion $0 \in \partial f(x) + \partial g(x)$, using only the resolvents of the individual operators $\partial f, \partial g$, and/or the individual operators $\partial f, \partial g$ themselves [7]. In convex analysis terms, this corresponds to splitting the proximal operator prox_{f+g} in terms of the simpler proximals prox_f and prox_g or gradients ∇f and ∇g. Two common splitting algorithms are the forward-backward splitting (FBS) and the Douglas-Rachford splitting (DRS), given as follows [7, 21]:

$$x^{k+1} = \mathrm{prox}_g(I - \nabla f)(x^k); \tag{FBS}$$

$$\begin{cases} x^{k+1} = \mathrm{prox}_f(y^k),\\ y^{k+1} = y^k + \mathrm{prox}_g(2x^{k+1} - y^k) - x^{k+1}. \end{cases} \tag{DRS}$$

One classical application of a splitting algorithm is the iterative thresholding and shrinkage algorithm (ISTA) for LASSO problems, where the fidelity f is quadratic, and the prior term is the ℓ_1 norm $g(x) = \|x\|_1$ [19, 9]. Applying the PnP framework to FBS and DRS, by replacing the prior proximal terms prox_g with a denoiser D_σ, gives the PnP-FBS and PnP-DRS methods.

$$\text{(PnP-FBS)} \qquad x^{k+1} = D_\sigma(I - \nabla f)(x^k);$$

$$\text{(PnP-DRS)} \qquad \begin{cases} x^{k+1} = \text{prox}_f(y^k), \\ y^{k+1} = y^k + D_\sigma(2x^{k+1} - y^k) - x^{k+1}. \end{cases}$$

Provable PnP results first arose by Chan et al. for the PnP-ADMM scheme, demonstrating fixed-point convergence under a bounded denoiser assumption $\|D_\sigma(x) - x\| \leq C\sigma^2$ [15]. Ryu et al. demonstrate convergence of the PnP-FBS algorithm when f is strongly convex and the denoiser residual $D_\sigma - I$ is Lipschitz with sufficiently small Lipschitz constant, as well as for PnP-DRS and PnP-ADMM in the case where $D_\sigma - I$ is Lipschitz with Lipschitz constant less than 1 [64]. Various works show fixed-point convergence of PnP-ADMM and PnP-FBS when f has Lipschitz gradient under an "averaged denoiser" assumption, where $(1-\theta)I + \theta D_\sigma$ is nonexpansive for some $\theta \in (0, 1)$, mainly using monotone operator theory [69, 70, 29]. Cohen et al. show fixed-point convergence of a relaxed PnP-FBS scheme when f has Lipschitz gradient under a demicontractive denoiser assumption, which is a strictly weaker condition than nonexpansiveness [17]. Sreehari et al. show convergence of PnP-ADMM to an implicitly defined convex function when the denoiser is nonexpansive and has symmetric gradient, by utilizing Moreau's theorem to characterize the denoiser as a proximal map of a convex function [66, 48]. In the case of nonexpansive linear denoisers, PnP-FBS and PnP-ADMM converge to fixed points of a closed-form convex optimization problem [51].

While plentiful, many of these convergence results impose restrictive or difficult-to-verify conditions on the denoisers D_σ. Instead of replacing the regularizing proximal operator prox_g with a denoiser, Hurault et al. and Cohen et al. instead consider applying FBS with the proximal operator on the fidelity term and a gradient step on the regularization, $x^{k+1} = \text{prox}_f(I - \nabla g)(x^k)$ [32, 16]. Replacing the regularization step with a denoiser $D_\sigma = I - \nabla g_\sigma$ results in the Gradient Step PnP (GS-PnP) algorithm $x^{k+1} = (\text{prox}_f \circ D_\sigma)(x^k)$. Using this parameterization, they show further that the fixed points of GS-PnP are stationary points of a particular (non-convex) function. Moreover, a follow-up work shows that a gradient-step denoiser of the form $D_\sigma = I - \nabla g_\sigma$ can be interpreted as a proximal step $D_\sigma = \text{prox}_{\phi_\sigma}$ [33]. Using this, they are able to achieve iterate convergence under KL-type conditions to a stationary point of a (non-convex) closed-form functional of the form (1.1).

The GS-PnP style schemes require that the gradient of the potential ∇g_σ is Lipschitz with Lipschitz constant $L < 1$. Methods of training neural networks with Lipschitz constraints include spectral regularization, adversarial training against Lipschitz bounds during training, or spline based architectures [64, 46, 22, 52]. Hurault et al. consider fine-tuning the DRUNet denoiser by using spectral regularization to enforce the Lipschitz gradient condition [33]. While it can be shown empirically that the Lipschitz constant is less than one locally, there is no theoretical guarantee, which can lead to occasional divergence. One possible way of remedying this is by averaging the denoiser with the identity operator, as remarked in [33]. This consists of replacing the denoiser $D_\sigma = I - \nabla g_\sigma$ with the relaxed $D_\sigma^\alpha := (1-\alpha)I + \alpha D_\sigma = I - \alpha\nabla g_\sigma$ for some $\alpha \in (0,1)$. We can rewrite the relaxed denoiser as $D_\sigma^\alpha = I - \nabla g_\sigma^\alpha$, where $g_\sigma^\alpha = \alpha g_\sigma$ has αL-Lipschitz gradient. Taking $\alpha < 1/L$ gives the appropriate contraction condition on g_σ^α and thus convergence of the associated PnP schemes [33, 31].

1.3. Quasi-Newton Methods. For minimizing a twice continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, a classical second-order method is Newton's method [54]:

$$x^{k+1} = x^k - (\nabla^2 f)^{-1}\nabla f(x^k), \tag{1.4}$$

where $\nabla^2 f$ is the Hessian of f. This can be interpreted as minimizing a local quadratic approximation

$$\hat{f}_k(y) = f(x^k) + \nabla f(x^k)^\top (y - x^k) + \frac{1}{2}(y - x^k)^\top \nabla^2 f(x^k)(y - x^k), \tag{1.5a}$$

$$x^{k+1} = \arg\min_y \hat{f}_k(y). \tag{1.5b}$$

Newton's method is able to achieve quadratic convergence rates with appropriate initialization and step-sizes [54]. However, the inverse of the Hessian may be computationally demanding, especially in high-dimensional applications such as image processing. Quasi-Newton (qN) methods propose to replace the inverse Hessian $(\nabla^2 f)^{-1}$ with (low-rank) approximations to the inverse Hessian, with notable examples including the Broyden-Goldfarb-Fletcher-Shanno (BFGS) algorithm, the David-Fletcher-Powell (DFP) formula, and the symmetric rank one method (SR1) [54].

Like Newton's method, quasi-Newton methods utilize the curvature information from the Hessian approximation to accelerate convergence, with applications in non-convex stochastic optimization, neural network training, and Riemannian optimization [13, 30, 75]. Classical theory gives asymptotic superlinear convergence under the Dennis-Moré condition, which states that the Hessian approximation converges to the Hessian at the minimum [20]. Non-asymptotic convergence of quasi-Newton methods is still an active area of research. BFGS and DFP have only recently been shown to have non-asymptotic superlinear convergence rates of $\mathcal{O}((1/k)^{k/2})$ when the objective function is strongly convex with Lipschitz continuous gradient, has Lipschitz continuous Hessian at the minimum, and satisfies a concordance condition [35, 61]. However, BFGS sees empirical success even when these conditions are not explicitly verified, including in the non-convex setting [41, 42]. Interestingly, certain accelerated proximal gradient methods can be interpreted as a proximal quasi-Newton method [55].

Variants of BFGS include limited memory BFGS (L-BFGS), stochastic BFGS, greedy BFGS, and sharpened BFGS [43, 34, 47, 65, 60]. Of these variants, the limited memory version is most suited to repeated iteration. Standard quasi-Newton methods continually update the Hessian approximation using all the previous iterates, leading to a linear per-iteration computational cost increase. L-BFGS instead utilizes only the last m iterates, where $m > 1$ is a user-specified parameter, typically chosen to be less than 50. Moreover, the Hessian need not be stored and/or computed at each iteration, as the method only relies on Hessian-vector products, which can be computed efficiently with two loop recursions [54].

To relate quasi-Newton methods to the PnP framework described previously, we would like to consider applying Newton-type methods for convex composite optimization, by replacing a proximal operator with a denoiser. Lee et al. consider the problem of minimizing

$$\varphi(x) = f(x) + g(x), \tag{1.6}$$

where $f(x)$ is a convex $\mathcal{C}^1$ function, and g is a possibly non-smooth convex regularizer [39]. For a symmetric positive definite matrix $B_k \approx \nabla^2 f(x^k)$, the proximal Newton-type search direction Δx^k, satisfying $x^{k+1} = x^k + t_k \Delta x^k$, is given as the minimizer of a local quadratic approximation on the smooth component $\hat{f}_k(y)$:

$$\hat{f}_k(y) = f(x^k) + \nabla f(x^k)^\top (y - x^k) + \frac{1}{2}(y - x^k)^\top B_k (y - x^k), \tag{1.7a}$$

$$\Delta x^k = \arg\min_d \hat{\varphi}_k(x^k + d) \coloneqq \hat{f}_k(x^k + d) + g(x^k + d). \tag{1.7b}$$

Define the *scaled proximal map* for a positive definite matrix B as in [39]:

$$\operatorname{prox}_g^B(x) \coloneqq \arg\min_{y \in \mathbb{R}^n} g(y) + \frac{1}{2}\|y - x\|_B^2, \tag{1.8}$$

where the B-norm is defined as $\|z\|_B^2 = z^\top B z$. For example, taking B to be the identity matrix results in the standard proximal map as defined in (1.2). The search direction (1.7b) has a closed form in terms of the scaled proximal map:

$$\Delta x^k = \operatorname{prox}_g^{B_k}(x - B_k^{-1} \nabla f(x^k)) - x^k. \tag{1.9}$$

With this search direction, appropriate step sizes and B_k, the proximal Newton-type methods are able to achieve similar convergence rates to Newton-type methods, achieving global convergence and local superlinear convergence. While the scaled proximal map allows for such analysis, it is not amenable to the PnP framework. For example, if we compute the Hessian approximation B_k using a BFGS-type approach, a naive approach of replacing $\operatorname{prox}_g^{B_k}$ with a denoiser would require a careful analysis of the interaction of B_k on the resulting regularization, and possibly require the denoiser to depend on B_k. Instead, we seek a proximal Newton-type method that utilizes only the unscaled proximal map, with possibly a scalar constant which can be easily interpreted as a regularization parameter controlling the strength of regularization.

In Section 2, we will detail a classical composite minimization algorithm that uses only the unscaled proximal map prox_g, as well as arbitrary descent steps that allow for Newton-type steps. We further extend the classical analysis from convex to weakly convex functions, inspired by the GS-PnP characterization of denoisers as proximal maps of weakly convex functions. In Section 3, we use this extension to propose the PnP-quasi-Newton (PnP-qN) method, further convergence and characterizing cluster points of the algorithm. In Section 4, we evaluate the proposed PnP-qN method with the quasi-Newton method given by L-BFGS, and compare it with other provable and non-provable PnP methods with comparable reconstruction quality.

2. Proximal Quasi-Newton.

In this section, we will first describe a classical algorithm for optimizing composite sums of a (possibly non-convex) smooth function and a (possibly non-smooth) convex function. We will then extend the analysis to allow for *weak convexity* instead of *convexity*. By replacing proximal terms with deep denoisers corresponding to proximal operators of weakly convex maps, we construct a Plug-and-Play scheme with convergence properties of the classical algorithm.

Let us work on the Euclidean domain $\mathbb{R}^n$. Let $\mathcal{C}^{1,1}_{L_f}$ denote the class of $\mathcal{C}^1$ functions $f : \mathbb{R}^n \to \mathbb{R}$ with L_f-Lipschitz gradient, and Γ_0 the class of proper, closed, and convex functions $g : \mathbb{R}^n \to \overline{\mathbb{R}}$. Consider a variational objective having the following form:

$$\varphi = f + g, \quad f \in \mathcal{C}^{1,1}_{L_f},\ g \in \Gamma_0. \tag{2.1}$$

We can consider f as the fidelity term and g as a regularization term. A prominent example from inverse problems is the quadratic fidelity loss $f(x; y) = \frac{1}{2}\|Ax - y\|^2$ for some linear forward operator $A : \mathbb{R}^n \to \mathbb{R}^m$ and observation $y \in \mathbb{R}^m$, where the norm is taken as the Euclidean norm.

2.1. MINFBE: Minimizing Forward-Backward Envelope.

We first detail a classical composite optimization algorithm for minimizing (2.1), which will serve as the base of our proposed PnP scheme. Moreover, we describe some of its convergence properties that transfer to the PnP framework. By constructing a smooth convex envelope function around the original objective φ, this envelope can be shown to have desirable properties such as sharing minimizers, smoothness, and being minorized and majorized by convex functions. By applying descent steps and proximal mappings in a particular fashion, the classical algorithm is able to obtain global objective convergence to critical points at a rate of $\mathcal{O}(1/k)$, local linear convergence if the function is locally strongly convex, and superlinear convergence when the descent steps are taken to be quasi-Newton with suitable assumptions [67].

For the problem (2.1), define the following expressions [67]:

$$l_\varphi(u, x) = f(x) + \langle \nabla f(x), u - x\rangle + g(u), \tag{2.2a}$$

$$T_\gamma(x) = \arg\min_u \left\{ l_\varphi(u, x) + \frac{1}{2\gamma}\|u - x\|^2 \right\} = \operatorname{prox}_{\gamma g}(x - \gamma \nabla f(x)), \tag{2.2b}$$

$$R_\gamma(x) = \gamma^{-1}(x - T_\gamma(x)), \tag{2.2c}$$

$$\varphi_\gamma(x) = \min_u \left\{ l_\varphi(u, x) + \frac{1}{2\gamma}\|u - x\|^2 \right\}. \tag{2.2d}$$

Here, l_φ is a local linearized decoupling of φ, T_γ can be interpreted as an FBS step (with step-size γ for $f + g$) and R_γ is a scaled residual or "gradient direction". Note that $x = T_\gamma(x) \Leftrightarrow x \in \operatorname{zer} \partial\varphi$, i.e. fixed points of T_γ correspond to critical points of φ. φ_γ is defined as the *forward-backward envelope* of φ. We further explicitly write the Moreau envelope for g:

$$g^\gamma(x) = \min_u \left\{ g(u) + \frac{1}{2\gamma}\|u - x\|^2 \right\} \tag{2.3a}$$

$$= g\left(\operatorname{prox}_{\gamma g}(x)\right) + \frac{1}{2\gamma}\|\operatorname{prox}_{\gamma g}(x) - x\|^2. \tag{2.3b}$$

With the above definitions, we have the following closed-form expressions for the forward-backward envelope:

$$\varphi_\gamma = f(x) + g(T_\gamma(x)) - \gamma\langle \nabla f(x), R_\gamma(x)\rangle + \frac{\gamma}{2}\|R_\gamma(x)\|^2 \tag{2.4a}$$

$$= f(x) - \frac{\gamma}{2}\|\nabla f(x)\|^2 + g^\gamma(x - \gamma\nabla f(x)). \tag{2.4b}$$

In fact, φ_γ has many desirable properties, such as sharing minimizers with φ, and having an easily computable derivative in terms of the Hessian of f.

Proposition 2.1 ([67, Sec 2]). *The following holds:*

i. $\varphi(z) = \varphi_\gamma(z)$ *for all* $\gamma > 0$, $z \in \operatorname{zer}\partial\varphi$*;*
ii. $\inf\varphi = \inf\varphi_\gamma$ *and* $\arg\min\varphi \subseteq \arg\min\varphi_\gamma$ *for* $\gamma \in (0, 1/L_f]$*;*
iii. $\arg\min\varphi = \arg\min\varphi_\gamma$ *for all* $\gamma \in (0, 1/L_f)$*.*

Suppose additionally that f *is* $\mathcal{C}^2$*. Then* φ_γ *is* $\mathcal{C}^1$ *and the gradient of* φ_γ *can be written as*

$$\nabla\varphi_\gamma(x) = \left(I - \gamma\nabla^2 f(x)\right) R_\gamma(x). \tag{2.5}$$

Moreover, if $\gamma \in (0, 1/L_f)$*, the set of stationary points of* φ_γ *equals* $\operatorname{zer}\partial\varphi$*.*

Assuming that we are able to compute both φ_γ and φ, Proposition 2.1(i) allows us to check whether we have converged to a stationary point of φ. Algorithm 2.1 is a classical forward-backward algorithm for optimizing the nonsmooth composite objective (2.1).

Algorithm 2.1 MINFBE [67]

Require: $x^0, \gamma_0 > 0, \xi \in (0,1), \beta \in [0,1), k \leftarrow 0$
1: **if** $R_{\gamma_k}(x^k) = 0$ **then**
2: stop
3: **end if**
4: Choose d^k s.t. $\langle d^k,\ \nabla\varphi_{\gamma_k}(x^k)\rangle \le 0$
5: Choose $\tau_k \ge 0$ and $w^k = x^k + \tau_k d^k$ s.t. $\varphi_{\gamma_k}(w^k) \le \varphi_{\gamma_k}(x^k)$
6: **if** $f(T_{\gamma_k}(w^k)) > f(w^k) - \gamma_k\langle\nabla f(w^k),\ R_{\gamma_k}(w^k)\rangle + \frac{(1-\beta)\gamma_k}{2}\|R_{\gamma_k}(w^k)\|^2$ **then**
7: $\gamma_k \leftarrow \xi\gamma_k$, goto 1
8: **end if**
9: $x^{k+1} \leftarrow T_{\gamma_k}(w^k)$
10: $\gamma_{k+1} \leftarrow \gamma_k$
11: $k \leftarrow k+1$, goto 1

In Algorithm 2.1, ξ is an Armijo backtracking parameter, while β is used to control the strictness of the descent condition in Step 6. For appropriately chosen γ, the condition in Step 6 never holds, as stated in the next lemma. Moreover, the step-sizes γ_k are bounded below by a constant in terms of σ, β and L_f. This guarantees that a step is always possible.

Lemma 2.2 ([67, Lem 3.1]). *Let* $(\gamma_k)_{k\in\mathbb{N}}$ *be the sequence of step-size parameters in Algorithm 2.1, and let* $\gamma_\infty = \min_{i\in\mathbb{N}}\gamma_i$*. Then for all* $k \ge 0$*,*

$$\gamma_k \ge \gamma_\infty \ge \min\{\gamma_0,\ \xi(1-\beta)/L_f\}.$$

The MINFBE algorithm can be interpreted as a descent step (Step 5) followed by a FBS step (Step 9). In particular, note that the descent direction d^k does not have to be the direction of steepest descent, which allows for more flexibility in the algorithm. By combining the two of these steps together, the algorithm achieves global convergence as well as local linear convergence. This algorithm enjoys the following convergence guarantees.

Definition 2.3 (Linear and Superlinear Convergence). *We say a sequence* $(x^k)_{k\in\mathbb{N}}$ *converges to* x_**;*

i. Q-linearly *with factor* $\omega \in [0,1)$ *if* $\|x^{k+1} - x_*\| \leq \omega \|x^k - x_*\|$ *for all* $k \geq 0$*;*

ii. Q-superlinearly *if* $\|x^{k+1} - x_*\| / \|x^k - x_*\| \to 0$.

The convergence is R-linear (R-superlinear) if $\|x^k - x_*\| \leq a_k$ *for some sequence* $(a_k)_{k\in\mathbb{N}}$ *s.t.* $a_k \to 0$ *Q-linearly (Q-superlinearly).*

Theorem 2.4 ([67, Thm 3.6, 3.7]). *Suppose that* f *is convex and that* φ *is coercive. In particular, suppose that the level set* $\{x \in \mathbb{R}^n \mid \varphi(x) \leq \varphi(x^0)\}$ *has diameter* R, $0 < R < \infty$*. Then for the sequences generated by Algorithm* 2.1*, either* $\varphi(x^0) - \inf \varphi \geq R^2/\gamma_0$ *and*

$$\varphi(x^1) - \inf \varphi \leq \frac{R^2}{2\gamma_0}, \tag{2.6}$$

or for any $k \in \mathbb{N}$*, it holds that*

$$\varphi(x^k) - \inf \varphi \leq \frac{2R^2}{k \min\{\gamma_0, \xi(1-\beta)/L_f\}}. \tag{2.7}$$

Suppose in addition that x_* *is a strong minimizer of* φ*, i.e. there exists a neighborhood* N *of* x_* *and* $c > 0$ *such that for any* $x \in N$*,*

$$\varphi(x) - \varphi(x_*) \geq \frac{c}{2}\|x - x_*\|^2.$$

Then for sufficiently large k*,* $(\varphi(x^k))_{k\in\mathbb{N}}$ *and* $(\varphi_{\gamma_k}(w^k))_{k\in\mathbb{N}}$ *converge Q-linearly to* $\varphi(x_*)$ *with factor* ω*, where*

$$\omega \leq \max\left\{\frac{1}{2}, 1 - \frac{c}{4}\min\{\gamma_0, \xi(1-\beta)/L_f\}\right\} \in [1/2, 1)\,,$$

and $(x^k)_{k\in\mathbb{N}}$ *converges R-linearly to* x_**. If* x_* *is also a strong minimizer of* φ_{γ_∞} *where* γ_∞ *is defined as in Lemma* 2.2*, then* $(\varphi(w^k))_{k\in\mathbb{N}}$ *also converges R-linearly to* x_**.*

In MINFBE, the initial descent step w^k can be chosen arbitrarily as long as the objective function decreases. Suppose now that the descent direction is chosen using a quasi-Newton method:

$$d^k = -B_k^{-1}\nabla\varphi_\gamma(x^k).$$

If B_k are positive definite, then d^k are valid search directions. Assuming that B_k satisfy the Dennis-Moré condition [54, 20], we can get superlinear convergence of the iterates.

Theorem 2.5 ([67, Thm 4.1]). *Fix $\gamma > 0$. Suppose that $\nabla\varphi_\gamma$ is strictly differentiable at a stationary point $x_* \in \operatorname{zer}\partial\varphi$, and that $\nabla^2\varphi_\gamma(x_*)$ is nonsingular. Let $(B_k)_{k\in\mathbb{N}}$ be a sequence of nonsingular $\mathbb{R}^{n\times n}$ matrices, and suppose the sequences*

$$w^k = x^k - B_k^{-1}\nabla\varphi_\gamma(x^k), \quad x^{k+1} = T_\gamma(w^k) \tag{2.8}$$

converge to x_*. If $x^k, w^k \notin \operatorname{zer}\partial\varphi$ for all $k \geq 0$ and the Dennis-Moré condition

$$\lim_{k\to\infty} \frac{\|(B_k - \nabla^2\varphi_\gamma(x^k))(w^k - x^k)\|}{\|w^k - x^k\|} = 0 \tag{2.9}$$

holds, then $(x^k)_{k\in\mathbb{N}}$ and $(w^k)_{k\in\mathbb{N}}$ converge Q-superlinearly to x_*.

If B_k are updated accordingly to the BFGS update step, then the updates as given in the previous theorem converge superlinearly to the minimum, under some additional assumptions on φ such as being convex with strong local minimum x_*, or satisfying a stronger Kurdyka-Łojasiewicz property at cluster points $\omega(x^0)$ [67, Thm 4.3]. Moreover, it can be shown that $\tau_k = 1$ is a valid step-size for sufficiently large k. For completeness, the BFGS update steps are given as below. Note that it is usually more practical to update the inverse Hessian approximation $H_k = B_k^{-1}$ [54].

$$s^k = w^k - x^k, \quad y^k = \nabla\varphi_\gamma(w^k) - \nabla\varphi_\gamma(x^k), \tag{2.10a}$$

$$B_{k+1} = \begin{cases} B_k + \frac{y^k y^{k\top}}{y^{k\top} s^k} - \frac{B_k s^k (B_k s^k)^\top}{s^{k\top} B^k s^k} & \text{if } \langle s^k, y^k\rangle > 0, \\ B_k & \text{otherwise} \end{cases}. \tag{2.10b}$$

$$H_{k+1} = \begin{cases} \left(I - \frac{s^k y^{k\top}}{y^{k\top} s^k}\right) H_k \left(I - \frac{y^k s^{k\top}}{y^{k\top} s^k}\right) + \frac{s^k s^{k\top}}{y^{k\top} s^k} & \text{if } \langle s^k, y^k\rangle > 0, \\ H_k & \text{otherwise} \end{cases}. \tag{2.10c}$$

2.2. Weakly-Convex Extension. Suppose now that g is not convex, but instead is M-weakly convex. Recall that a function $g(x)$ is M-weakly convex if $g + M\|x\|^2/2$ is convex. For a M-weakly convex function g, we have for all x, y and $z \in \partial g(y)$ (where ∂g denotes the Clarke subdifferential of g),

$$g(x) \geq g(y) + \langle z, x - y\rangle - \frac{M}{2}\|x - y\|^2, \tag{2.11a}$$

$$g(tx + (1-t)y) \leq tg(x) + (1-t)g(y) + \frac{M}{2}t(1-t)\|x - y\|^2. \tag{2.11b}$$

In the following Section 3, we will model the proposed denoiser $D_\sigma = \operatorname{prox}_g$ as the proximal operator of a weakly convex function. In particular, a gradient step denoiser $D_\sigma = I - \nabla g_\sigma$ with contractive ∇g_σ is the proximal operator of a weakly convex function [31]. We can extend the classical convex analysis to this case as well, albeit with a smaller allowed γ.

To transfer the results from the previous section to the case where g is weakly convex, we are required to check that the function values at the MINFBE iterates are non-increasing. As we will show in the following proposition, this is still the case for sufficiently small γ. Many properties of the forward-backward envelope still hold, and we are still able to attain global convergence and superlinear local convergence, subject to the Dennis-Moré condition (2.9).

Proposition 2.6. *For all $x \in \mathbb{R}^n$, $\gamma > 0$,*

i. $\varphi_\gamma(x) \le \varphi(x) - \frac{\gamma - M\gamma^2}{2}\|R_\gamma(x)\|^2$;
ii. $\varphi(T_\gamma(x)) \le \varphi_\gamma(x) - \frac{\gamma}{2}(1-\gamma L_f)\|R_\gamma(x)\|^2$ *for all* $\gamma > 0$;
iii. $\varphi(T_\gamma(x)) \le \varphi_\gamma(x)$ *for all* $\gamma \in (0, 1/L_f]$.

Proof. **(i).** By the optimality condition in (2.2b), we have

$$R_\gamma(x) - \nabla f(x) \in \partial g(T_\gamma(x)).$$

By (2.11a), we have

$$\begin{aligned} g(x) &\ge g(T_\gamma(x)) + \langle R_\gamma(x) - \nabla f(x), x - T_\gamma(x)\rangle - \frac{M}{2}\|x - T_\gamma(x)\|^2 \\ &= g(T_\gamma(x)) - \gamma\langle \nabla f(x), R_\gamma(x)\rangle + \gamma\|R_\gamma(x)\|^2 - \frac{M\gamma^2}{2}\|R_\gamma(x)\|^2. \end{aligned}$$

Adding $f(x)$ to both sides and applying (2.4a) gives the result.

(ii), (iii). The proof is identical to that in [67, Prop 2.2], requiring only the Lipschitz convexity of ∇f. ∎

Proposition 2.7. *Suppose $\gamma - M\gamma^2 \ge 0$, or equivalently $\gamma \in [0, 1/M]$. Then the following hold:*

i. $\varphi_\gamma(z) = \varphi(z)$ *for all* $z \in \operatorname{zer} \partial\varphi$;
ii. $\inf \varphi = \inf \varphi_\gamma$ *and* $\arg\min \varphi \subseteq \arg\min \varphi_\gamma$ *for* $\gamma \in (0, 1/L_f]$;
iii. $\arg\min \varphi = \arg\min \varphi_\gamma$ *for* $\gamma \in (0, 1/L_f)$.

Proof. **(i).** Proposition 2.6(i) combined with the condition $\gamma - M\gamma^2 \ge 0$ shows $\varphi_\gamma(x) \le \varphi(x)$. If $z \in \operatorname{zer} \partial\varphi$, then $z = T_\gamma(z)$, and Proposition 2.6(ii) reads $\varphi(z) \le \varphi_\gamma(z)$.

(ii), (iii). Identical to [67, Prop 2.3]. ∎

With weakly convex functions, we are still able to provide a lower bound on the γ such that the condition in Step 6 of Algorithm 2.1 does not hold, removing the need to reduce step-sizes. The proof relies only on the Lipschitz constant of ∇f and does not require convexity of g. However, we require that $\gamma - M\gamma^2 \ge 0$. In practice, the denoisers we use have $M < 1/2$, which allows for any $\gamma \in (0, 1)$.

Lemma 2.8. *Suppose g is weakly convex. If $0 < \gamma < \min\{(1-\beta)/L_f, 1/M\}$, then the condition in Step 6 in Algorithm 2.1 never holds. Moreover, this implies MINFBE iterations satisfy $\gamma_k \ge \gamma_\infty \ge \min\{\gamma_0, \xi(1-\beta)/L_f, 1/M\} > 0$ for all k.*

Proof. Suppose $0 < \gamma < \min\{(1-\beta)/L_f, 1/M\}$, and for contradiction that the condition in Step 6 holds. Then there exists some w such that

$$f(T_\gamma(w)) > f(w) - \gamma\langle \nabla f(w), R_\gamma(w)\rangle + \frac{(1-\beta)\gamma}{2}\|R_\gamma(w^k)\|^2.$$

Adding $g(T_\gamma(w))$ to both sides and considering (2.4a), this becomes

$$\varphi(T_\gamma(w)) > \varphi_\gamma(w) - \frac{\beta\gamma}{2}\|R_\gamma(w)\|^2.$$

But from Proposition 2.6(ii), we also have

$$\begin{aligned}\varphi(T_\gamma(w)) &\le \varphi_\gamma(w) - \frac{\gamma}{2}(1-\gamma L_f)\|R_\gamma(w)\|^2 \\ &\le \varphi_\gamma(w) - \frac{\beta\gamma}{2}\|R_\gamma(w)\|^2,\end{aligned}$$

where the second inequality follows from $\gamma < (1-\beta)/L_f$, giving a contradiction. The second part holds since $(\gamma_k)_{k\in\mathbb{N}}$ is a non-increasing sequence. ■

Remark 2.9. While $\gamma < 1/M$ is not strictly needed for the proof of the above lemma, this requirement is needed for convergence in future results.

The following theorem characterizes the convergence of the functional φ, which relies on the non-increasing condition of Step 5 in Algorithm 2.1. This is an analogue of [67, Prop 3.4].

Theorem 2.10. *Suppose $0 < \gamma_0 < 1/M$. Then the MINFBE iterations satisfy the following:*
- *i.* $\varphi(x^{k+1}) \le \varphi(x^k) - \frac{\beta\gamma_k}{2}\|R_{\gamma_k}(w^k)\|^2 - \frac{\gamma_k - M\gamma_k^2}{2}\|R_{\gamma_k}(x^k)\|^2$;
- *ii. Either the sequence $\|R_{\gamma_k}(x^k)\|$ is square-summable, or $\varphi(x^k) \to \inf\varphi = -\infty$ and the set $\omega(x^0)$ of cluster points of the sequence $(x^k)_{k\in\mathbb{N}}$ is empty.*
- *iii.* $\omega(x^0) \subseteq \operatorname{zer}\partial\varphi$;
- *iv. If $\beta > 0$, then either the sequence $\|R_{\gamma_k}(w^k)\|$ is square-summable and every cluster point of $(w^k)_{k\in\mathbb{N}}$ is critical, or $\varphi_{\gamma_k}(w^k) \to \inf\varphi = -\infty$ and $(w^k)_{k\in\mathbb{N}}$ has no cluster points.*

Proof. **(i).** Recalling $x^{k+1} = T_{\gamma_k}(w^k)$,

$$\varphi(x^{k+1}) \le \varphi_{\gamma_k}(w^k) - \frac{\beta\gamma_k}{2}\|R_{\gamma_k}(w^k)\|^2$$

$$\le \varphi_{\gamma_k}(x^k) - \frac{\beta\gamma_k}{2}\|R_{\gamma_k}(w^k)\|^2 \tag{2.12}$$

$$\le \varphi(x^k) - \frac{\beta\gamma_k}{2}\|R_{\gamma_k}(w^k)\|^2 - \frac{\gamma_k - M\gamma_k^2}{2}\|R_{\gamma_k}(x^k)\|^2, \tag{2.13}$$

where the first and second inequalities come from Step 6 and 5 in Algorithm 2.1 respectively, and the final inequality is Proposition 2.6(i).

(ii)-(iv). We follow [67] with minor modifications. Let $\varphi_* = \lim_{k\to\infty}\varphi(x^k)$, which exists as $(\varphi(x^k))_{k\in\mathbb{N}}$ is monotone by (i) and $\gamma_k - M\gamma_k^2 \ge 0$. If $\varphi_* = -\infty$, then $\inf\varphi = -\infty$. By properness and lower semi-continuity of φ, as well as the monotonicity of $\varphi(x^k)$, no cluster points of $(x^k)_{k\in\mathbb{N}}$ exist. If instead $\varphi_* > -\infty$, by telescoping (2.13),

$$\frac{1}{2}\sum_{i=0}^{k}\gamma_i\left(\beta\|R_{\gamma_i}(w^i)\|^2 + (1-\gamma_i M)\|R_{\gamma_i}(x^i)\|^2\right) \le \varphi(x^0) - \varphi(x^{k+1}) \le \varphi(x^0) - \varphi_*. \tag{2.14}$$

Since γ_k is uniformly bounded below by Lemma 2.8, we have square summability of $\|R_{\gamma_k}(x^k)\|$, showing (ii).

By square summability, $R_{\gamma_k}(x^k) \to 0$. Moreover, the functions $R_{\gamma_k} = R_{\gamma_\infty}$ are constant for sufficiently large k, and R_{γ_∞} is continuous by continuity of the proximal operator and of ∇f.

Therefore, any cluster point $z \in \omega(x^k)$ has $R_{\gamma_\infty}(x^{k_j}) \to R_{\gamma_\infty}(z) = 0$ for some subsequence $x^{k_j} \to z$. Thus $z = T_{\gamma_\infty}(z) \Rightarrow z \in \operatorname{zer} \partial\varphi$, showing (iii).

If $\beta > 0$, for sufficiently large k such that $\gamma_k = \gamma_\infty$, the following chain of inequalities holds:

$$\varphi_{\gamma_k}(w^{k+1}) \le \varphi_{\gamma_k}(x^{k+1}) = \varphi_{\gamma_k}(T_k(w^k)) \le \varphi_{\gamma_k}(w^k). \tag{2.15}$$

The first inequality comes from Step 5, the equality from Step 9, and the final inequality from Proposition 2.6. The monotonicity of $\varphi_{\gamma_k}(w^k)$ for sufficiently large k allows for a similar argument to hold for the w^k sequence, giving (iv). ■

Convergence results can also be extended to the weakly convex case. In particular, the following theorem shows the convergence of the residuals between each step.

Theorem 2.11 (Global Residual Convergence). *Suppose $0 < \gamma_0 \le 1/(2M)$, and let $c = \min\{\gamma_0, \xi(1-\beta)/L_f, 1/M\} > 0$ be the lower bound for γ_∞. The MINFBE iterations satisfy*

$$\min_{i \le k} \|R_{\gamma_i}(x^i)\|^2 \le \frac{2}{k+1} \frac{\varphi(x^0) - \inf \varphi}{c - Mc^2}. \tag{2.16}$$

If in addition $\beta > 0$, then we also have

$$\min_{i \le k} \|R_{\gamma_i}(w^i)\|^2 \le \frac{2}{k+1} \frac{\varphi(x^0) - \inf \varphi}{\beta c}. \tag{2.17}$$

Proof. As in [67, Thm 3.5]. If $\inf \varphi = -\infty$, there is nothing to prove, so suppose otherwise that $\inf \varphi > -\infty$. Considering (2.14) along with $(\gamma_k)_{k \in \mathbb{N}}$ being nonincreasing implies

$$\frac{(k+1)(\gamma_k - M\gamma_k^2)}{2} \min_{i \le k} \|R_{\gamma_i}(x^i)\|^2 + \frac{(k+1)\beta\gamma_k}{2} \min_{i \le k} \|R_{\gamma_i}(w^i)\|^2 \le \varphi(x^0) - \inf \varphi. \tag{2.18}$$

Now note that $\gamma - M\gamma^2$ is increasing for $\gamma < 1/(2M)$, so $\gamma_k - M\gamma_k^2$ is lower bounded by $c - Mc^2 > 0$. Rearranging yields both inequalities. ■

To obtain convergence of the objective similar to Theorem 2.4, it is insufficient for g to be weakly convex. We can alternatively utilize the KL property, which is a useful and general property satisfied by a large class of functions, including semialgebraic functions [4]. Moreover, it can be used to show convergence in the absence of other regularity conditions such as convexity [5, 10, 33].

Definition 2.12 (KL Property [5, 10]). *Suppose $\varphi : \mathbb{R}^n \to \overline{\mathbb{R}}$ is proper and lower semicontinuous. φ satisfies the* Kurdyka-Łojasiewicz (KL) property *at a point x_* in $\operatorname{dom} \partial\varphi$ if there exists $\eta \in (0, +\infty]$, a neighborhood U of x_* and a continuous concave function $\Psi : [0, \eta) \to [0, +\infty)$ such that:*

1. $\Psi(0) = 0$*;*
2. Ψ *is* $\mathcal{C}^1$ *on* $(0, \eta)$*;*
3. $\Psi'(s) > 0$ *for* $s \in (0, \eta)$*;*
4. *For all $u \in U \cap \{\varphi(x_*) < \varphi(u) < \varphi(x_*) + \eta\}$, we have*

$$\varphi'(\varphi(u) - \varphi(x_*)) \operatorname{dist}(0, \partial\varphi(u)) \ge 1.$$

We say that φ is a KL function if the KL property is satisfied at every point of $\operatorname{dom} \partial\varphi$.

Utilizing the KL property, we are able to show that the iterates generated by MINFBE are sufficiently well-behaved, and hence converge. Moreover, from Theorem 2.10, we have that the iterates converge to critical points of the non-convex objective φ. Under the PnP scheme, this will correspond to convergence to critical points of some function determined by the denoiser.

Theorem 2.13. *Suppose that f satisfies the KL condition and g is semialgebraic, and both f and g are bounded from below. Suppose further that there exist constants $\bar{\tau}, c > 0$ such that $\tau_k < \bar{\tau}$ and $\|d^k\| \leq c\|R_{\gamma_k}(x^k)\|$, $\beta > 0$, and that φ is coercive or has compact level sets. Then the sequence of iterates $(x^k)_{k\in\mathbb{N}}$ is either finite and ends with $R_{\gamma_k}(x^k) = 0$, or converges to a critical point of φ.*

Proof. Deferred to the supplementary material. The proof is very similar to that in [67, Thm 3.9, Appendix 4]. ∎

The crux of using the MINFBE method is that we are able to incorporate Newton-type steps into the iterations. Since we are able to get convergence to a critical point from the previous theorem, we are in a position to apply the next theorem to show superlinear convergence in a neighborhood of a minimizer.

Theorem 2.14. *Suppose that f is continuously differentiable with L_f-Lipschitz gradient and g is M-weakly convex. Let $\gamma = \gamma_\infty$ as in Lemma 2.8. Suppose the search directions are chosen as*

$$d^k = -B_k^{-1}\nabla\varphi_\gamma(x^k),$$

the step-sizes in Step 5 are chosen with $\tau_k = 1$ tried first, and B_k satisfy the Dennis-Moré condition (2.8). Suppose further that the iterates $(x^k)_{k\in\mathbb{N}}$, $(w^k)_{k\in\mathbb{N}}$ converge to a critical point x_* at which $\nabla\varphi_\gamma$ is continuously differentiable with $\nabla^2\varphi_\gamma(x_*) \succ 0$. Then $(x^k)_{k\in\mathbb{N}}$ and $(w^k)_{k\in\mathbb{N}}$ converge Q-superlinearly to x_*.

Proof. The proof is nearly identical to [67, Thm 4.1]. If γ_g is M-weakly convex, then for $\gamma < 1/M$, $u \mapsto \left(g(u) + \frac{1}{2\gamma}\|u - x\|^2\right)$ is strongly convex. Thus $\operatorname{prox}_{\gamma g}$ is 1-Lipschitz [59]. The rest of the proofs of Thm 4.1 and 4.2 of [67] follows as usual. ∎

This shows superlinear convergence instead of linear convergence in the case where the critical point is a strong local minimum, i.e. it is locally strongly convex. Note the differentiability condition in the second part can be dropped if f and g are both $\mathcal{C}^2$. Moreover, assuming either φ is convex and x_* is a strong local minimum, or φ satisfies a stronger KL inequality, these conditions indeed hold if B_k is updated according to the BFGS scheme [67, Thm 4.3].

3. PnP-qN: Deep Denoiser Extension.

To convert Algorithm 2.1 to the PnP framework, we consider replacing the proximal step in (2.2b) with a denoiser. In particular, we consider the gradient-step denoiser setup in [33]. Let the denoiser D_σ be given by

$$D_\sigma = I - \nabla g_\sigma, \tag{3.1a}$$

$$g_\sigma = \frac{1}{2}\|x - N_\sigma(x)\|^2, \tag{3.1b}$$

where g_σ is a $\mathcal{C}^2$ function with L-Lipschitz gradient with $L < 1$. Note the subscript in g_σ represents a denoising strength, as opposed to the forward-backward envelope of g as we will define for our problem later. The mapping $N_\sigma(x)$ takes the form of a $\mathcal{C}^2$ neural network, allowing for the computation of g_σ explicitly. Under these assumptions, the denoiser D_σ takes the form of a proximal mapping of a weakly convex function, as stated in the next proposition.

Proposition 3.1 ([31, Prop 1]). *$D_\sigma(x) = \operatorname{prox}_{\phi_\sigma}(x)$, where ϕ_σ is defined by*

$$\phi_\sigma(x) = g_\sigma(D_\sigma^{-1}(x)) - \frac{1}{2}\|D_\sigma^{-1}(x) - x\|^2 \tag{3.2}$$

if $x \in \operatorname{Im}(D_\sigma)$, and $\phi_\sigma(x) = +\infty$ otherwise. Moreover, ϕ_σ is $\frac{L}{L+1}$-weakly convex.

This proposition allows us to take the weak convexity constant required in the previous section as $M = L/(L+1)$. Since $L < 1$, we have $M < 1/2$. This result can be thought of a slight extension of the fact that a function f is a proximal operator of some proper convex l.s.c. function φ, if and only if it is a subgradient of a convex l.s.c. function ψ and f is nonexpansive [27, 48].

Suppose that $\gamma_k = \gamma > 0$ is fixed in the MINFBE iterations, satisfying the conditions in Lemma 2.8. Consider making the substitution with ϕ_σ defined as in Proposition 3.1, targeting $\varphi = f + g$:

$$\gamma g = \phi_\sigma. \tag{3.3}$$

The FBS step $T_\gamma(x) = \operatorname{prox}_{\gamma g}(x - \gamma\nabla f(x))$ thus becomes, using $D_\sigma = \operatorname{prox}_{\phi_\sigma}$,

$$T_\gamma(x) = D_\sigma(x - \gamma\nabla f(x)). \tag{3.4}$$

This will target the objective function $\varphi(x) = f(x) + g(x) = f(x) + \phi_\sigma(x)/\gamma$. To iterate Algorithm 2.1 with this substitution, we need to evaluate φ_γ. Recalling (2.4b), we can instead evaluate the Moreau envelope g^γ. By definition (2.3b) and the substitution (3.3), we have:

$$\begin{aligned}
g^\gamma(y) &\overset{(2.3b)}{=} g(\operatorname{prox}_{\gamma g}(y)) + \frac{1}{2\gamma}\|\operatorname{prox}_{\gamma g}(y) - y\|^2 \\
&\overset{(3.3)}{=} \frac{1}{\gamma}\phi_\sigma(D_\sigma(y)) + \frac{1}{2\gamma}\|D_\sigma(y) - y\|^2 \\
&\overset{(3.2)}{=} \frac{1}{\gamma}g_\sigma(D_\sigma^{-1}(D_\sigma(y))) - \frac{1}{2\gamma}\|D_\sigma^{-1}(D_\sigma(y)) - D_\sigma(y)\|^2 + \frac{1}{2\gamma}\|D_\sigma(y) - y\|^2 \\
&= \frac{1}{\gamma}g_\sigma(y).
\end{aligned}$$

Using this substitution, we obtain the Plug-and-Play scheme PnP-MINFBE, detailed in Algorithm 3.1. We have a closed form for the forward-backward envelope of φ, as well as some other expressions essential for iterating MINFBE, given by:

$$\varphi(x) = f(x) + \frac{1}{\gamma}\phi_\sigma(x), \tag{3.5a}$$

$$\varphi_\gamma(x) = f(x) - \frac{\gamma}{2}\|\nabla f(x)\|^2 + \frac{1}{\gamma}g_\sigma(x - \gamma\nabla f(x)), \tag{3.5b}$$

$$\nabla\varphi_\gamma(x) = (I - \gamma\nabla^2 f)R_\gamma(x), \tag{3.5c}$$

$$\varphi(x^{k+1}) = f(x^{k+1}) + \frac{1}{\gamma}\left(g_\sigma(w^k - \gamma\nabla f(w^k)) - \|w^k - \gamma\nabla f(w^k) - T_\gamma(w^k)\|^2/2\right). \tag{3.5d}$$

Algorithm 3.1 PnP-MINFBE

Require: $x^0, \gamma < \min\{\gamma_0, (1-\beta)/L_f, 1/M\}, \beta \in [0, 1), k \leftarrow 0$

1: **if** $R_{\gamma_k}(x^k) = 0$ **then**
2: stop
3: **end if**
4: Choose d^k s.t. $\langle d^k,\ \nabla\varphi_\gamma(x^k)\rangle \leq 0$
5: Choose $\tau_k \geq 0$ and $w^k = x^k + \tau_k d^k$ s.t. $\varphi_\gamma(w^k) \leq \varphi_\gamma(x^k)$
6: $x^{k+1} \leftarrow D_\sigma(w^k - \gamma\nabla f(w^k))$
7: $k \leftarrow k + 1$, goto 1

To compute the search direction d^k at each step, we can use a quasi-Newton method to approximate the inverse Hessian of φ_γ. While a closed form exists for $\nabla^2\varphi_\gamma$, such as in [67, Thm 2.10], it requires the Jacobian of the denoiser D_σ, rendering methods requiring the Hessian computationally intractable due to the dimensionality of our problems. Therefore, we resort to a BFGS-like algorithm using the differences and secants

$$s^k = w^k - x^k,\ y^k = \nabla\varphi_\gamma(w^k) - \nabla\varphi_\gamma(x^k).$$

In particular, we will use the L-BFGS method due to the memory restrictions imposed by using images for our experiments. This can be implemented using a two-loop recursion, using only the last m secants computed [54]. We additionally impose a safeguard to reject updating the Hessian approximation if the secant condition $\langle s^k, y^k\rangle > 0$ is not satisfied. For completeness, we write the two-loop recursion for L-BFGS in Algorithm 3.2. The initial (inverse) Hessian approximations are chosen as $H_0^k = c_k I$ as in [54], given by

$$c_k = \frac{\langle s^{k-1}, y^{k-1}\rangle}{\langle y^{k-1}, y^{k-1}\rangle}.$$

Utilizing the results from the previous section, we can show the following convergence results for PnP-MINFBE (Algorithm 3.1) and PnP-LBFGS (Algorithm 3.3).

Corollary 3.2. *Suppose that f is $\mathcal{C}^1$ and KL with L_f-Lipschitz gradient, g_σ is $\mathcal{C}^2$ and semi-algebraic with L_g-Lipschitz gradient with $L_g < 1$. Assume further that $\gamma < 1/(2M)$ is chosen as in Lemma 2.8 such that $\gamma = \gamma_\infty$, and there exist $\bar\tau, c > 0$ such that $\tau_k \leq \bar\tau$ and $\|d^k\| \leq c\|R_\gamma(x^k)\|$. Then the PnP-MINFBE iterations of Algorithm 3.1 satisfy the following:*

Algorithm 3.2 L-BFGS [54]

Require: $m > 0$, secants $(s^i)_{i=k-m}^{k-1}$, differences $(y^i)_{i=k-m}^{k-1}$, initial Hessian guesses $(H_0^k)_{k\in\mathbb{N}}$

$q \leftarrow \nabla\varphi_\gamma(x^k)$
$\rho_i \leftarrow 1/\langle y^i, s^i\rangle$ for $i = k-1, k-2, ..., k-m$
for $i = k-1, k-2, ..., k-m$ **do**
  $\alpha_i \leftarrow \rho_i\langle s^i, q\rangle$
  $q \leftarrow q - \alpha_i y^i$
end for
$r \leftarrow H_0^k q$
for $i = k-m, k-m+1, ..., k-1$ **do**
  $\beta \leftarrow \rho_i\langle y^i, r\rangle$
  $r \leftarrow r + (\alpha_i - \beta)s^i$
end for
stop with $B_k^{-1}\nabla\varphi_\gamma(x^k) = H^k\nabla\varphi_\gamma(x^k) = r$

Algorithm 3.3 PnP-LBFGS

Require: $x^0, \gamma < \min\{(1-\beta)/L_f, 1/M\}, \beta \in [0,1), k \leftarrow 0$

if $R_{\gamma_k}(x^k) = 0$ **then**
  stop
end if
Compute $d^k \leftarrow -B_k^{-1}\nabla\varphi_\gamma(x^k)$ using L-BFGS (c.f. Algorithm 3.2) with differences and secants $(s^i, y^i)_{i=k-m}^{k-1}$.
Choose $\tau_k \in [0,1]$ and $w^k = x^k + \tau_k d^k$ s.t. $\varphi_\gamma(w^k) \le \varphi_\gamma(x^k)$
$x^{k+1} \leftarrow D_\sigma(w^k - \gamma\nabla f(w^k))$
$s^k \leftarrow w^k - x^k,\ y^k \leftarrow \nabla\varphi_\gamma(w^k) - \nabla\varphi_\gamma(x^k)$
$k \leftarrow k+1$, goto 1

i. $\varphi(x^k)$ decreases monotonically;
ii. The residuals $R_\gamma(x^k)$ converge to zero at a rate $\mathcal{O}(1/\sqrt{k})$;
iii. If the iterates are bounded, then the iterates are either finite or converge to a critical point of $\varphi = f + \frac{1}{\gamma}\phi_\sigma$. Moreover, $\varphi = \varphi_\gamma$ at these critical points.
iv. If furthermore $d^k = -B_k^{-1}\nabla\varphi_\gamma(x^k)$ and the B_k satisfy the Dennis-Moré condition (2.8), then the x^k and w^k converge superlinearly to x_*.

Proof. **(i), (ii).** Follows from Theorems 2.10 and 2.11. **(iii).** By the Tarski-Siedenberg theorem [5], compositions and inverses of semi-algebraic mappings are semi-algebraic. Therefore D_σ and D_σ^{-1} are semi-algebraic (on their domain), and hence so is ϕ_σ. Therefore,

$$\varphi = f + \frac{1}{\gamma}\phi_\sigma$$

is a KL function. Moreover, φ_γ is also a KL function. So we have convergence by Theorem 2.13. The final part follows from Proposition 2.7. **(iv).** Follows from Theorem 2.14. ■

Table 1: Hyperparameters for PnP-LBFGS.

	Deblur			SR		
σ	2.55	7.65	12.75	2.55	7.65	12.75
α	0.5	0.5	0.7	0.5	0.5	0.5
γ	1					
β	0.01					
λ	1	1	1	4	1.5	1
σ_d/σ	1	0.75	0.75	2	1	0.75

Table 2: Hyperparameters for PnP-$\hat{\alpha}$PGD.

	Deblur			SR		
σ	2.55	7.65	12.75	2.55	7.65	12.75
α	0.6	0.8	0.85	1	1	1
L_f	1			0.25		
λ	$(\alpha+1)/(\alpha L_f)$					
$\hat{\alpha}$	$1/(\lambda L_f)$					
σ_d/σ	1.5	1	1	2	2	2

Remark 3.3. An essential part of the classical proof relies on the fact that $\tau = 1$ will eventually always be accepted in MINFBE, under a Newton-type descent direction choice. During numerical testing, we observed that the Armijo search for τ was only occasionally necessary when the image is being optimized, with at most 10 line searches required before converging.

In our case, f will be a quadratic fidelity term of the form $f(x) = \|Ax - y\|^2/2$ for some linear operator A and measurement y. This is semi-algebraic and hence KL, and moreover trivially bounded below. From (3.1b), we additionally have that g_σ is bounded below. Since N_σ will take the form of a neural network which is a composition of semi-algebraic operations and arithmetic operations, g_σ will also be semi-algebraic. Therefore, we can apply Corollary 3.2 and get convergence to critical points of the associated function $\varphi = f + \frac{1}{\gamma}\phi_\sigma$.

4. Experiments.

In this section, we consider the application of the proposed PnP-LBFGS method, given by Algorithm 3.3, with a pre-trained denoiser to image deblurring and super-resolution. We use the pretrained Lipschitz-constrained proximal denoiser given in [33]. The (gradient-step) denoiser takes the form (3.1), where N_σ is a neural network based on the DRUNet architecture [77]. The Lipschitz constraint on ∇g_σ is enforced by applying a penalty on the spectral norm of $\nabla^2 g_\sigma$ during training. While this spectral constraint affects the performance of the end-to-end denoiser, it provides sufficient conditions for convergence in the context of PnP, in particular, convergence to a critical point of a closed-form functional.

The datasets we consider for image reconstruction are the CBSD68, CBSD10 and set3c datasets[1], containing images of size 256×256 with three color channels and pixel intensity values in $[0, 255]$ [45]. The forward operators corresponding to the considered reconstruction problems of deblurring and super-resolution are linear, and we can write the fidelity term as $f(x) = \lambda\|Ax - y\|^2/2$, where A is the degradation operator, y is the degraded image, and λ is a regularization parameter. For reconstruction, y will be taken as $y = Ax_{\text{true}} + \varepsilon$, where x_{true} is the ground-truth image and the noise ε is pixel-wise Gaussian with standard deviations $\sigma \in \{2.55, 7.65, 12.75\}$ corresponding to 1%, 3%, and 5% noise (relative to the maximum pixel intensity value), respectively. The underlying optimization problems corresponding to fixed points of PnP-MINFBE thus take the form (as in (3.5a)):

$$\min_x \varphi(x) = \frac{\lambda}{2}\|Ax - y\|^2 + \frac{1}{\gamma}\phi_\sigma, \tag{4.1}$$

where $\gamma \leq \min\{(1-\beta)/L_f, 1/2M\}$ as in Lemma 2.8 and Theorem 2.11. In this case, f is $\mathcal{C}^2$, and we can easily compute the derivative of the forward-backward envelope using (3.5c).

The methods we compare against are PnP methods with similar convergence guarantees, namely $\mathcal{O}(1/\sqrt{k})$ residual convergence and a KL-type iterate convergence [33]. Our analysis additionally shows superlinear convergence to minima with positive-definite Hessian using Newton's directions. Although we can not verify whether the Hessian approximation B_k obtained via L-BFGS satisfies the Dennis-Moré condition for superlinear convergence, we will empirically demonstrate faster convergence in terms of both time and iteration count compared to the competing methods.

The PnP methods that we will compare against are the PnP-PGD, PnP-DRS, PnP-DRSdiff and PnP-$\hat{\alpha}$PGD methods [33, 31]. Here PGD stands for proximal gradient descent, DRS for Douglas-Rachford splitting, DRSdiff for DRS with differentiable fidelity terms, and $\hat{\alpha}$PGD for $\hat{\alpha}$-relaxed PGD. The update rules corresponding to the chosen PnP methods for comparison are as follows:

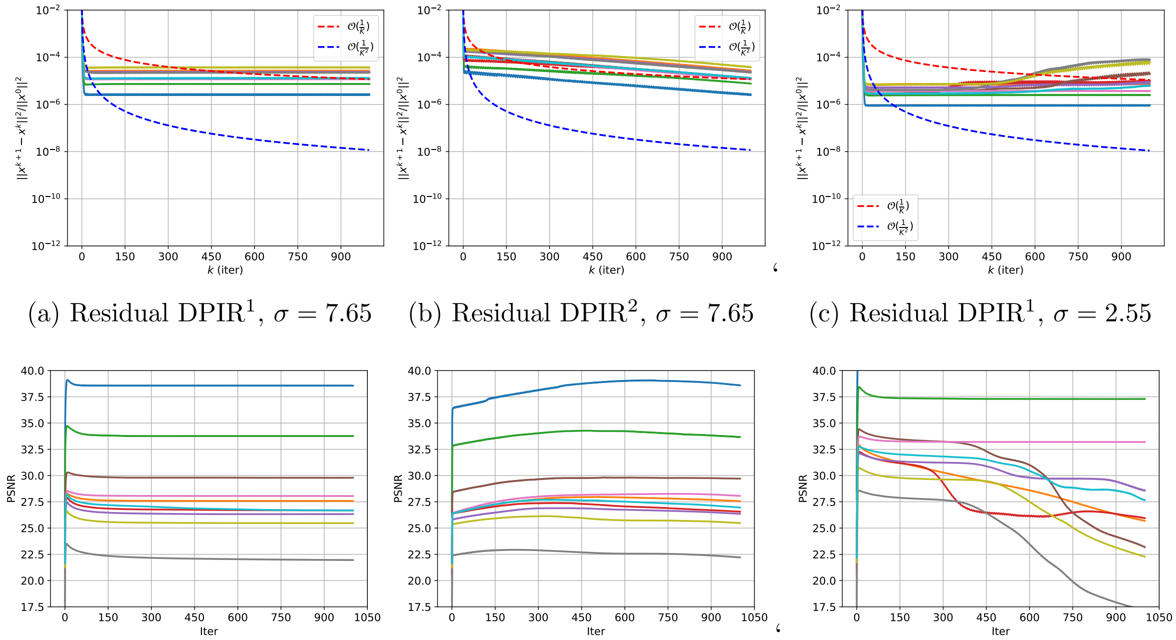
$$\text{(PnP-PGD)} \quad \begin{cases} z^{k+1} = x^k - \lambda\nabla f(x^k) \\ x^{k+1} = D_\sigma(z^{k+1}) \end{cases}$$

$$\text{(PnP-DRSdiff)} \quad \begin{cases} y^{k+1} = \operatorname{prox}_{\lambda f}(x^k) \\ z^{k+1} = D_\sigma(2y^{k+1} - x^k) \\ x^{k+1} = x^k + (z^{k+1} - y^{k+1}) \end{cases}$$

$$\text{(PnP-DRS)} \quad \begin{cases} y^{k+1} = D_\sigma(x^k) \\ z^{k+1} = \operatorname{prox}_{\lambda f}(2y^{k+1} - x^k) \\ x^{k+1} = x^k + (z^{k+1} - y^{k+1}) \end{cases}$$

$$\text{(PnP-}\hat{\alpha}\text{PGD)} \quad \begin{cases} q^{k+1} = (1-\hat{\alpha})y^k + \hat{\alpha}x^k \\ x^{k+1} = D_\sigma(x^k - \lambda\nabla f(q^{k+1})) \\ y^{k+1} = (1-\hat{\alpha})y^k + \hat{\alpha}x^{k+1} \end{cases}$$

[1] https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/



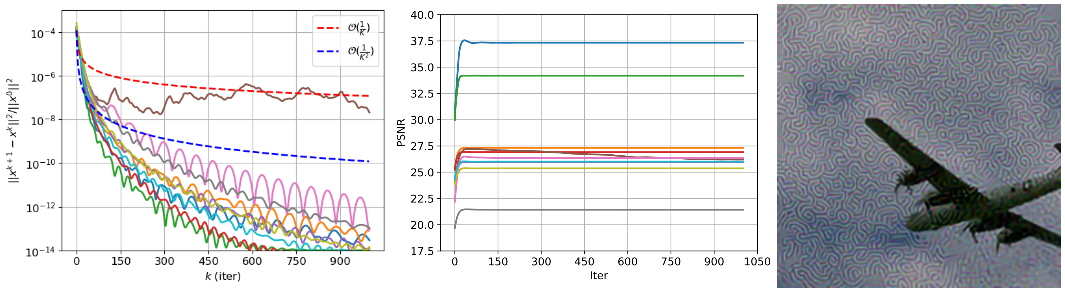
(a) Residual DPIR^1, $\sigma = 7.65$ (b) Residual DPIR^2, $\sigma = 7.65$ (c) Residual DPIR^1, $\sigma = 2.55$

(d) PSNR DPIR^1, $\sigma = 7.65$ (e) PSNR DPIR^2, $\sigma = 7.65$ (f) PSNR DPIR^1, $\sigma = 2.55$

Figure 1: Performance of DPIR measured in terms of residual $\|x^{k+1} - x^k\|^2/\|x^0\|^2$ and PSNR for deblurring with noise levels $\sigma = 2.55, 7.65$, applied with two different denoiser strength regimes. Each curve corresponds to one of the 10 images from the CBSD10 dataset. DPIR^1 has denoiser strength decreased from 49 to σ over 8 iterations for deblurring, and extended with $\sigma_d = \sigma$ for following iterations. DPIR^2 has denoiser strength decreased from 49 to σ over 1000 iterations. We observe that both methods have decreasing PSNR at later iterations and non-converging residual, and further that DPIR diverges for small noise levels.

4.1. Hyperparameter and Denoiser Choices. The hyperparameters for the proposed PnP-LBFGS and the existing PnP-$\hat{\alpha}$PGD methods are as in Tables 1 and 2, respectively, chosen via grid search to maximize the PSNR over the set3c dataset for the respective image reconstruction problems. The hyperparameter grid for PnP-LBFGS is given in the subsequent subsections, while the grid for PnP-$\hat{\alpha}$PGD is given below. For the denoiser in our experiment, we use the pre-trained network N_σ as in [33].

The convergence conditions for PnP-PGD and PnP-DRSdiff are that g_σ has L-Lipschitz gradient for some $L < 1$, and directly using the denoiser D_σ maintains theoretical convergence. For PnP-DRS, the condition needs to be strengthened to $L < 1/2$. In this case, the denoiser is replaced with an averaged denoiser of the form $(I+D_\sigma)/2 = I - \frac{1}{2}\nabla g_\sigma$, which gives convergence results but changes the underlying optimization problem. For PnP-LBFGS and PnP-$\hat{\alpha}$PGD, we use an averaged denoiser $D_\sigma^\alpha = I - \alpha \nabla g_\sigma$ which appears to have better performance, with the relaxation parameter α chosen as in Tables 1 and 2. As remarked in the introduction, adding the relaxation parameter α means that the effective Lipschitz constant of the potential



(a) Residual PnP-FISTA (b) PSNR PnP-FISTA (c) Failure example

Figure 2: Residual $\|x^{k+1} - x^k\|^2/\|x^0\|^2$ and PSNR for PnP-FISTA applied to super-resolution with noise level $\sigma = 7.65$. Each curve corresponds to one of the 10 images from the CBSD10 dataset. Using the parameters of PnP-LBFGS, which should resolve any Lipschitz constraint issues, has the same divergence issue. PnP-FISTA sometimes fails, leading to images with artifacts as seen in subfigure (c).

gradient $\alpha\nabla g_\sigma$ is αL, which alleviates divergence issues when $L > 1$. In this case, $D_\sigma^\alpha = \mathrm{prox}_{\phi_\sigma^\alpha}$ for some weakly convex ϕ_σ^α, and the previous computations hold with g_σ replaced with αg_σ.

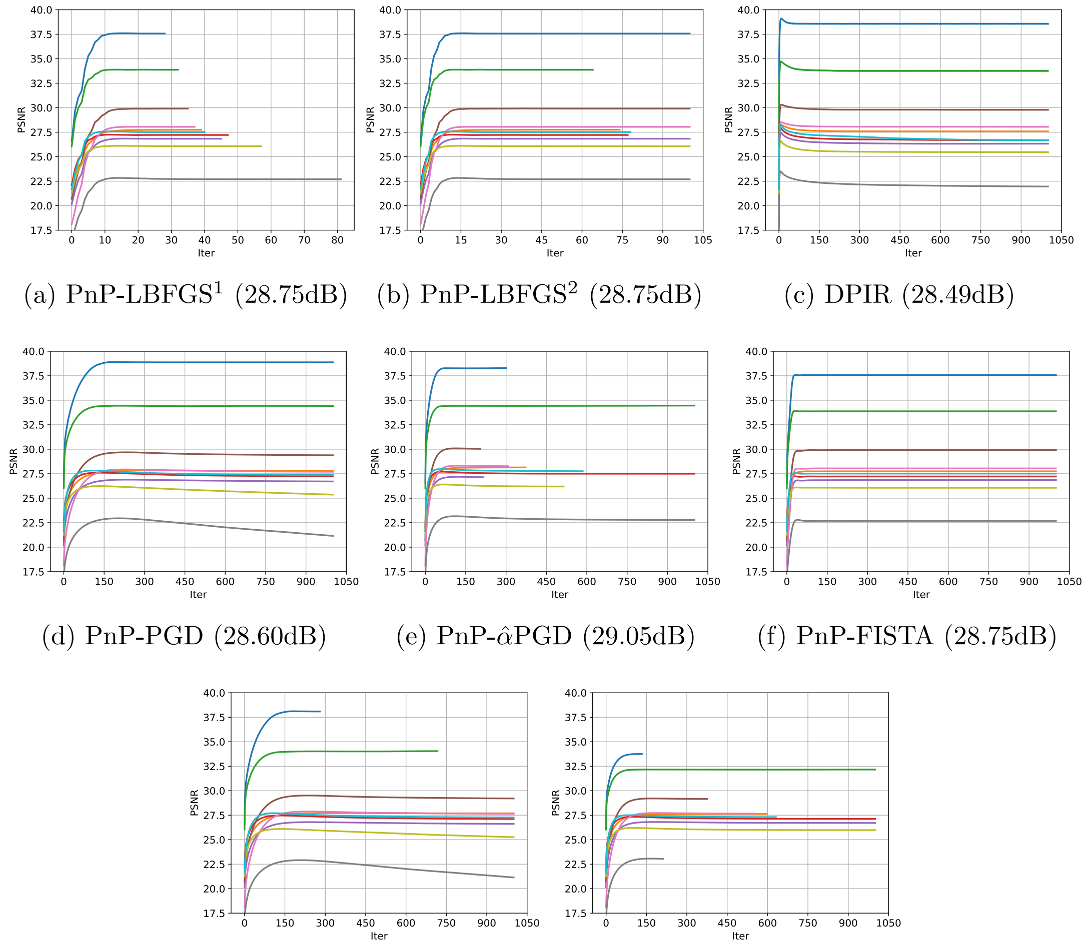
For the parameters of the relaxed PnP-$\hat{\alpha}$PGD algorithm, we perform a grid search as in [31]. To obtain the values of the denoiser averaging parameter α and the denoiser strength σ_d, we do a grid search for the set3c dataset with $\alpha \in \{0.6, 0.7, 0.8, 0.85, 0.9, 1.0\}$ and $\sigma_d/\sigma \in \{0.5, 0.75, 1.0, 1.5, 2.0\}$, where the noise level is $\sigma = 7.65$. The main difficulty in finding these hyperparameters is the dependence between α and σ_d, leading to poor reconstructions for many of these values. Given the denoiser averaging parameter α, the other hyperparameters of PnP-$\hat{\alpha}$PGD are given by $\lambda = \frac{\alpha+1}{\alpha L_f}, \hat{\alpha} = \frac{1}{\lambda L_f}$.

For the Lipschitz constant, we take $L_f = 1$ for deblurring and $L_f = 1/4$ for super-resolution with $s_{sr} = 2, 3$, as in Subsections 4.3 and 4.4. It appears approximating $L_f = 1$ for super-resolution or $L_f = 1/9 = 1/s_{sr}^2$ for $s_{sr} = 3$ results in divergence, indicating sensitivity to their hyperparameters. We find the best values to be as in Table 2, with the grid search taken to maximize the PSNR over the set3c dataset. We additionally employ a stopping criterion based on the Lyapunov functional that PnP-$\hat{\alpha}$PGD minimizes, with the same sensitivity as PnP-DRS and PnP-DRSdiff [31].

The regularization parameter λ for the underlying optimization problem is restricted for PnP-LBFGS in a manner similar to PnP-PGD and PnP-DRS (but not PnP-DRSdiff). For PnP-PGD and PnP-DRS, one condition for convergence is that $\lambda L_f < 1$ [33]. However, for PnP-LBFGS, Lemma 2.8 gives the condition that $\gamma < (1-\beta)/(\lambda L_f)$, targeting stationary points of

$$\varphi(x) = \frac{\lambda}{2}\|Ax - y\|^2 + \frac{1}{\gamma}\phi_\sigma.$$

We note that as λ increases, the allowed γ decreases, which correspondingly increases the smallest allowed coefficient $1/\gamma$ of the prior ϕ_σ at the same rate as λ. This puts an upper



(a) PnP-LBFGS[1] (28.75dB) (b) PnP-LBFGS[2] (28.75dB) (c) DPIR (28.49dB)

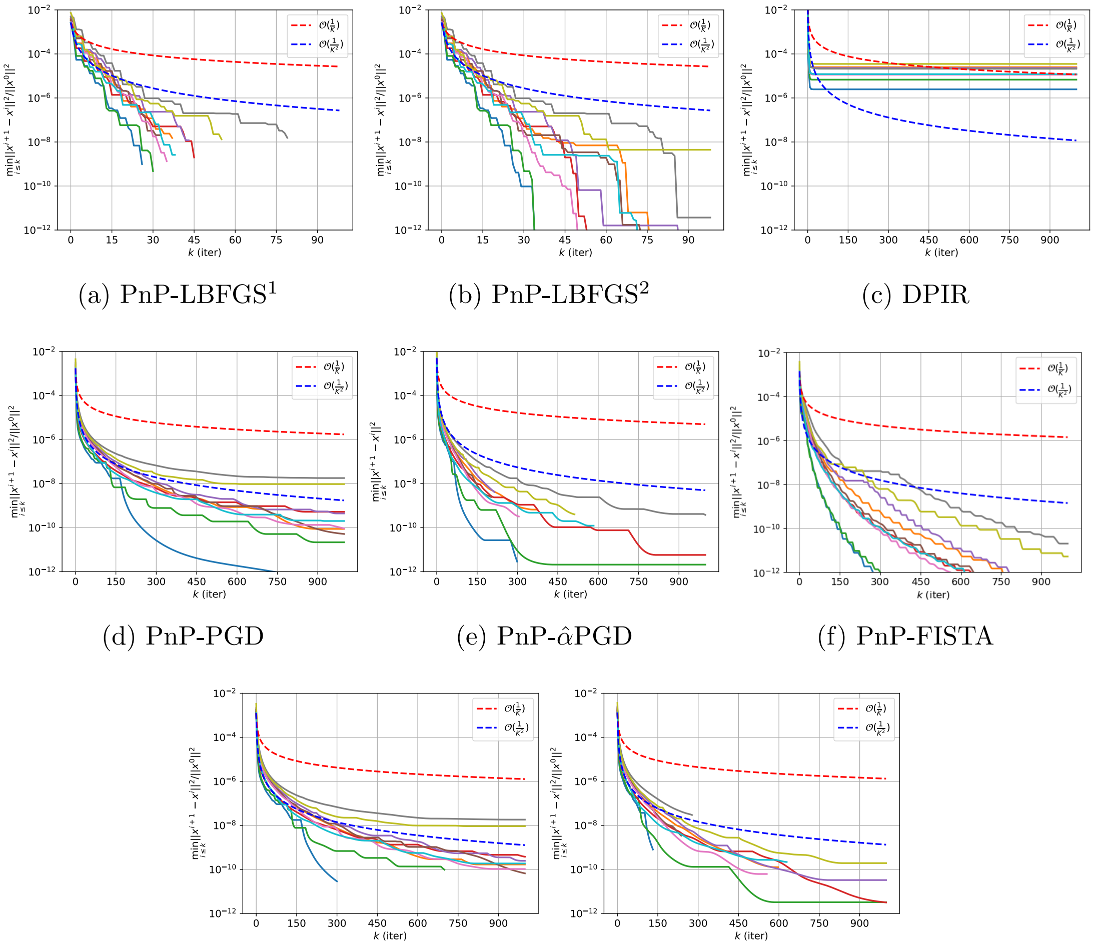
(d) PnP-PGD (28.60dB) (e) PnP-$\hat{\alpha}$PGD (29.05dB) (f) PnP-FISTA (28.75dB)

(g) PnP-DRSdiff (28.61dB) (h) PnP-DRS (28.80dB)

Figure 3: Convergence of the PSNRs for deblurring, with the average dB in brackets. Each curve corresponds to one of the 10 images from the CBSD10 dataset. Note that the scale of (a) is 10 times smaller than the other curves, terminating at 100 instead of 1000. PnP-LBFGS and PnP-DRS have generally more stable convergence, which can be attributed to the smaller Lipschitz constant of $I - D_\sigma$. PnP-LBFGS[1] also converges in much fewer iterations than the compared methods. The average PSNR between PnP-LBFGS with the two stopping criteria differ by only 0.0013dB.

bound on the ratio between the fidelity term and the regularization term, which may be restrictive for low-noise applications.

The memory length for LBFGS was chosen to be $m = 20$, with a maximum of 100 iterations per image. The denoiser D_σ^α is chosen with denoising strength σ_d similar to that used for PnP-DRS as in [33]. By using different denoising strengths, we are able to further



(a) PnP-LBFGS[1] (b) PnP-LBFGS[2] (c) DPIR

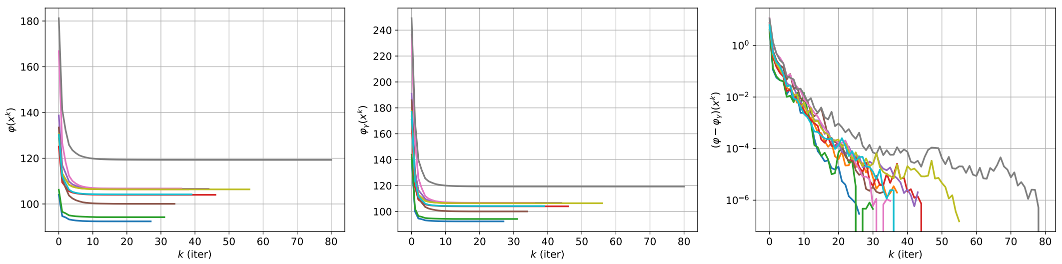
(d) PnP-PGD (e) PnP-$\hat{\alpha}$PGD (f) PnP-FISTA

(g) PnP-DRSdiff (h) PnP-DRS

Figure 4: Convergence of the residuals $\min_{i\leq k}\|x^{i+1}-x^i\|^2/\|x^0\|^2$ of the various methods for deblurring. Each curve corresponds to one of the 10 images from the CBSD10 dataset, evaluated with the first blur kernel and $\sigma = 7.65$. Note that the x-axis scale of (a) is 10 times smaller than the other curves, terminating at 100 instead of 1000.

control regularization along with the scaling parameter λ. The step-sizes τ_k are chosen using an Armijo line search starting from $\tau_k = 1$, and multiplying by 0.5 if the φ_γ decrease condition in Step 5 of Algorithm 3.3 is not met [3, 8].

We additionally introduce a stopping criterion based on the differences between consecutive iterates of the envelope $\varphi_\gamma(x^{k+1}) - \varphi_\gamma(x^k) < 10^{-5}$, as well as the envelope and objective $\varphi(x^k) - \varphi_\gamma(x^k) < 5 \times 10^{-5}$, where we stop if at least one criterion is met for 5 iterations in a row. We note that while the criteria can be strengthened, there is minimal change in the optimization result. We label PnP-LBFGS with the envelope-based stopping criterion as



(a) $\varphi(x_k)$ (b) $\varphi_\gamma(x_k)$ (c) $\varphi(x_k) - \varphi_\gamma(x_k)$

Figure 5: Evolution of the objective φ, forward-backward envelope φ_γ, and their difference $\varphi - \varphi_\gamma$ for deblurring with PnP-LBFGS^1. These values are equal at the true minima, i.e., $\varphi_\gamma(x_*) = \varphi(x_*)$. Each curve corresponds to one of the 10 images from the CBSD10 dataset, evaluated with the first blur kernel and $\sigma = 7.65$.

PnP-LBFGS^1. For completeness, we also consider the stopping criterion when the relative difference between consecutive function values of φ is less than 10^{-8}. We label PnP-LBFGS with the objective change stopping criterion as PnP-LBFGS^2. The PnP-LBFGS algorithms with the two stopping criteria are labeled with superscripts, as PnP-LBFGS^1 and PnP-LBFGS^2, respectively. We further use PnP-LBFGS without superscripts to refer to both methods together, which share their parameters.

All implementations were done in PyTorch, and the experiments were performed on an AMD EPYC 7352 CPU and a Quadro RTX 6000 GPU with 24GB of memory [56]. The code for our experiments are publicly available[2].

4.2. PnP Methods Without Convergence Guarantees. For further comparison, we additionally consider two non-provable PnP methods, namely DPIR [77] and PnP-FISTA [38]. DPIR is based on the half-quadratic splitting, which splits prox_{f+g} into alternating prox_f and prox_g steps, and further replaces prox_g with a denoising step D_{σ_k} in the spirit of PnP. PnP-FISTA is based on the fast iterative shrinkage-thresholding algorithm, which arises by applying a Nesterov-style acceleration to the forward-backward splitting [38, 37]. We note that neither of these methods correspond to critical points of functions in the existing literature.

$$\text{(DPIR)} \qquad \begin{cases} \alpha_k = \hat\lambda \sigma^2/\sigma_k^2, \\ x_{k+1} = \mathrm{prox}_{f/2\alpha_k}(z_k), \\ z_{k+1} = D_{\sigma_k}(x_k). \end{cases}$$

$$\text{(PnP-FISTA)} \qquad \begin{cases} x_k = D_\sigma(y_k - \lambda \nabla f(y_k)), \\ t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}, \\ y_{k+1} = x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}). \end{cases}$$

[2]https://github.com/hyt35/Prox-qN

4.2.1. DPIR. To improve the performance, DPIR uses a decreasing noise regime as well as image transformations during iteration [77, Sec. 4.2]. To extend past eight iterations, we consider using the log-scale noise from $\sigma_d = 49$ to $\sigma_d = \sigma$ over 8 and 24 iterations for deblurring and super-resolution respectively, as recommended in the DPIR paper [77, Sec. 5.1.1, 5.2]. The scaling for the proximal term is determined by a scaling parameter $\hat{\lambda}$, which was chosen to be $\hat{\lambda} = 0.23$ in the original work. Figure 1 shows that while DPIR achieves state-of-the-art performance in the low iteration regime, the PSNR begins to drop when HQS is extended past the number of iterations used in the original DPIR paper [32]. Moreover, DPIR appears to have poor performance in the low noise regime for the following image reconstruction experiments. In the following experiments, we consider DPIR with the suggested 8 and 24 iterations for deblurring and super-resolution respectively, as well as extending up to 1000 iterations to check the convergence behavior.

4.2.2. PnP-FISTA. The denoiser parameters for PnP-FISTA are considered to be either the parameters for PnP-LBFGS or PnP-PGD. Proofs for PnP schemes such as PnP-PGD or PnP-DRS generally rely on classical monotone operator theory, and showing that the denoiser satisfies the necessary assumptions. However, proofs of convergence of FISTA depend heavily on the convexity of the problem [9, 14], and non-convex proofs additionally require techniques or conditions such as adaptive backtracking [24, 55] or quadratic growth conditions [6]. These techniques and conditions are difficult to convert and verify in the PnP regime, which translates to difficulties in showing convergence of the associated PnP-FISTA schemes.

In the following experiments, we run the DPIR and PnP-FISTA methods for 1000 iterations unless stated otherwise to verify the convergence behavior. Figures 1 and 2 additionally demonstrate some common modes of divergence for DPIR and PnP-FISTA, with DPIR failing for low noise levels and PnP-FISTA failing with artifacts.

4.3. Deblurring. For deblurring, 10 blur kernels were used, including eight camera shake kernels, a 9×9 uniform kernel, and a 25×25 Gaussian kernel with standard deviation $\sigma_{\text{blur}} = 1.6$ [40, 33]. Visualizations of the kernels can be found in the supplementary material. The blurring operator A corresponds to convolution with circular boundary conditions. In this case, the transpose $A^\top$ can be easily implemented using a transposed convolution with circular boundary conditions. The blurring operator was previously scaled to have $\|A^\top A\|_{\text{op}} \approx 0.96$, which was verified using a power iteration. Thus, ∇f is approximately 0.96λ-Lipschitz.

We chose hyperparameters of PnP-LBFGS following a grid search maximizing the PSNR on the set3c dataset. The parameter grids are $\alpha \in \{0.5, 0.7, 0.9, 1.0\}$, $\lambda \in \{0.8, 0.9, 1.0\}$, $\gamma \in \{0.8, 0.85, 0.9, 1.0\}$, and $\sigma_d/\sigma \in \{0.5, 0.75, 1.0, 1.5, 2.0\}$. Note that this choice obeys $\gamma < \min\{(1-\beta)/L_f, 1/(2M)\}$, since φ_σ is at most 1/2-weakly convex. We observe empirically that the step-size $\tau = 1$ is also a valid descent almost all of the time, verifying the claim that is required to prove the superlinear convergence as remarked in Remark 3.3. The underlying optimization problems are slightly different for PnP-LBFGS and PnP-PGD: for PnP-PGD, the fidelity regularization is chosen to be $\lambda = 0.99$, and the iterates converge to cluster points of $\varphi_{\text{PnP-PGD}}$:

$$\varphi_{\text{PnP-LBFGS}} = \frac{1}{2}\|Ax - y\|^2 + \phi_\sigma^\alpha, \quad \varphi_{\text{PnP-PGD}} = \frac{0.99}{2}\|Ax - y\|^2 + \phi_\sigma.$$

We observe in Table 3 that the PnP-PGD and PnP-DRSdiff converge to very similar results

Table 3: Table of average PSNR (dB) comparing existing provable and non-provable PnP methods evaluated on the CBSD68 dataset compared to the proposed PnP-LBFGS methods. The time shown is the average reconstruction time per image. The PnP-LBFGS[1] method is significantly faster per image due to the faster convergence compared to the other provable PnP methods.

σ	2.55	7.65	12.75	Time (s)
PnP-LBFGS[1]	31.19	27.95	26.61	5.80
PnP-LBFGS[2]	31.17	27.78	26.61	9.55
PnP-PGD	30.57	27.80	26.61	25.93
PnP-DRSdiff	30.57	27.78	26.61	22.72
PnP-DRS	31.54	28.07	26.60	19.26
PnP-$\hat{\alpha}$PGD	31.52	28.15	26.74	15.66
PnP-FISTA	30.24	27.15	26.60	24.32
DPIR (iter 10^3)	27.40	27.58	26.46	19.62
DPIR (iter 8)	32.01	28.34	26.86	0.55

since they both minimize the same underlying functional. However, the PnP iterations sometimes do not converge, as demonstrated by the steadily decreasing PSNR in subfigures (d) and (g) of Figure 3. This can be attributed to the Lipschitz constant of g_σ being greater than 1 at these iterates. The use of the averaged denoiser D_σ^α in PnP-DRS and PnP-LBFGS reduces divergence, where we see convergence for these images as well. We generally observe that PnP-$\hat{\alpha}$PGD has the best performance in terms of PSNR, which can be attributed to the larger allowed value of λ. Nonetheless, we observe significantly faster convergence for PnP-LBFGS compared to the other methods to comparable PSNR values for each test image.

Comparing with the non-provable PnP methods, we observe in Figure 3 that PnP-FISTA converges to the same PSNR as PnP-LBFGS on CBSD10, but has a worse performance when averaged over all CBSD68 images in Table 3. This can be attributed to divergence of the method for denoisers where the Lipschitz constant of ∇g_σ is greater than 1. DPIR instead reaches its peak in the first couple of iterations, before decreasing to the fixed point as iterated by the denoiser with the final denoising strength $\sigma_d = \sigma$. This results in worse performance of DPIR at iteration 10^3 as compared to iteration 8, demonstrating the non-convergence and the current gap in performance between provable PnP and non-provable PnP.

Figure 3 and Figure 4 additionally demonstrate the difference between the stopping criteria. The stopping criteria of PnP-LBFGS[1] is sufficient for convergence to a reasonable PSNR, and allows for much earlier stopping. PnP-LBFGS[2] stops after more iterates and demonstrates the significantly faster convergence of the residuals compared to the other considered PnP methods. Moreover, Figure 5 shows the convergence curves of the objective φ and forward-backward envelope φ_γ, which rapidly converge to the same value, verifying Proposition 2.1.

4.4. Super-resolution. For super-resolution, we consider the forward operator with scale $s_{sr} \in \{2, 3\}$ as $A = SK : \mathbb{R}^{n\times n} \to \mathbb{R}^{\lfloor n/s_{sr}\rfloor \times \lfloor n/s_{sr}\rfloor}$, which is a composition of a downsam-

(a) Ground Truth (b) PnP-LBFGS[1] (29.78dB) (c) DPIR iter 100 (29.31dB) (d) DPIR iter 8 (30.13dB) (e) PnP-$\hat{\alpha}$PGD (30.00dB)

(f) Corrupted (g) PnP-PGD (28.68dB) (h) PnP-FISTA (29.58dB) (i) PnP-DRSdiff (28.66dB) (j) PnP-DRS (29.39dB)

Figure 6: Deblurring visualization using starfish image, with each method limited to a maximum of 100 iterations. Experiments are run with additive Gaussian noise $\sigma = 7.65$. PnP-LBFGS[1] converges within the first 100 iterations, while the other PnP algorithms take longer to converge. Since the result of PnP-LBFGS[1] and PnP-LBFGS[2] are nearly identical, we show only PnP-LBFGS[1]. DPIR starts to decrease in PSNR after 8 iterations, leading to slightly worse performance.

pling operator $S : \mathbb{R}^{n\times n} \to \mathbb{R}^{\lfloor n/s_{sr}\rfloor \times \lfloor n/s_{sr}\rfloor}$ and a circular convolution $K : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$. The convolutions K are Gaussian blur kernels with blur strength given by standard deviations $\sigma_{\text{blur}} = \{0.7, 1.2, 1.6, 2.0\}$ as in [77, 33]. For the PnP-LBFGS parameters, we chose hyperparameters maximizing the PSNR using a grid search on the set3c dataset over the following ranges: $\alpha \in \{0.5, 0.7, 0.9, 1.0\}$, $\lambda \in \{1.0, 2.0, 3.0, 4.0\}$, $\gamma \in \{0.8, 0.85, 0.9, 1.0\}$, and $\sigma_d/\sigma \in \{0.5, 0.75, 1.0, 1.5, 2.0\}$.

The Hessian $\nabla^2 f = \lambda A^\top A = \lambda K^\top S^\top S K$ is easily available, as $S^\top S : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ is a mask operator comprised of setting pixels with index not in $(s_{sr}\mathbb{Z})^2$ to zero, and $K^\top$ is a transposed convolution with circular boundary conditions. Note that on the image manifold, $S^\top S$ is approximately $1/s_{sr}^2$-Lipschitz, as we set $(s_{sr}^2 - 1)/s_{sr}^2$ of the pixels to zero. With K being approximately 1-Lipschitz, we have that $A^\top A$ is approximately $1/s_{sr}^2$-Lipschitz.

The PnP-LBFGS parameters are $\beta = 0.01, \gamma = 1$, and $\lambda = 2, 1.5, 1$ for noise levels $\sigma = 2.55, 7.65, 12.75$ respectively. We can take these values of λ since $L_f \approx 1/s_{sr}^2 \leq 1/4$ and $\gamma = 1$ still obeys $\gamma < \min\{(1-\beta)/L_f, 1/(2M)\}$. The underlying functionals are as follows:

$$\varphi_{\text{PnP-LBFGS}} = \frac{\lambda_{\text{LBFGS}}}{2}\|Ax - y\|^2 + \phi_\sigma^\alpha, \quad \varphi_{\text{PnP-PGD}} = \frac{0.99}{2}\|Ax - y\|^2 + \phi_\sigma.$$

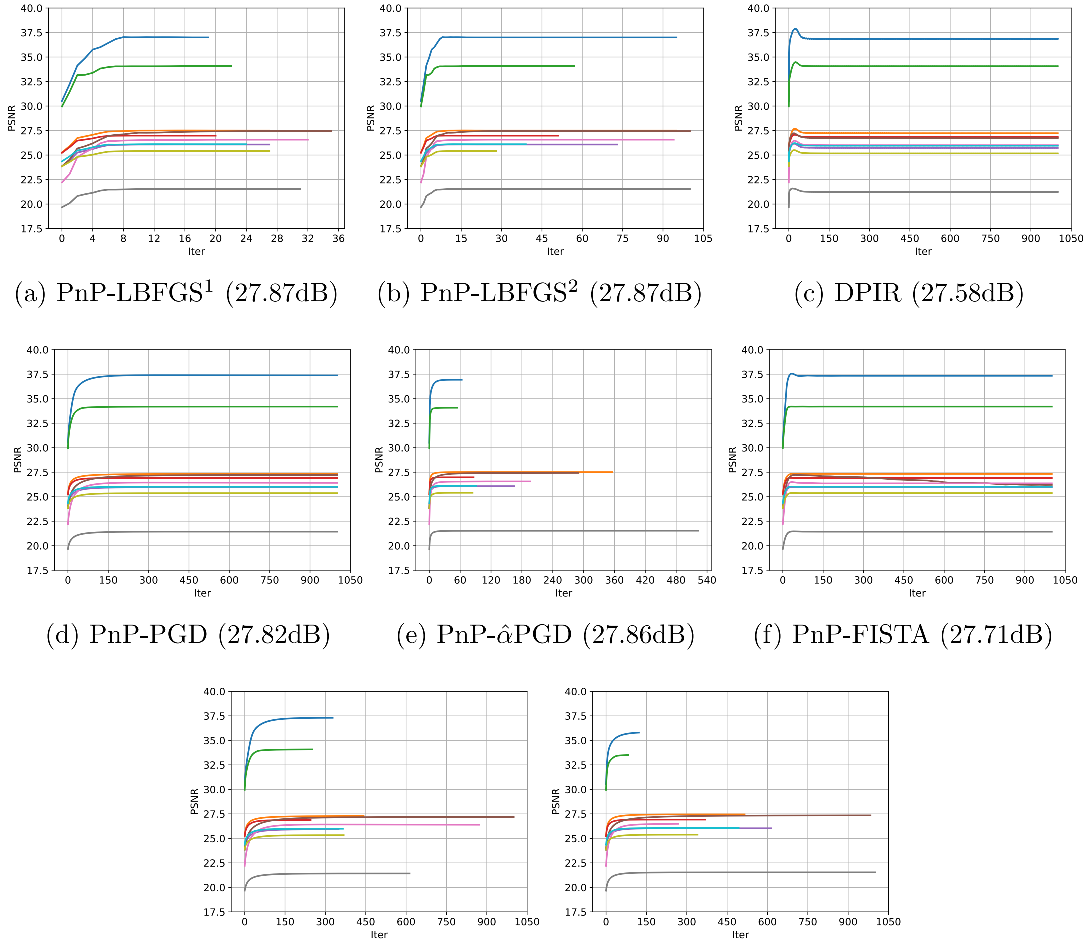
Table 4: Table of averaged PSNR (dB) corresponding to the competing PnP methods evaluated on the CBSD68 dataset for super-resolution, as compared with the proposed PnP-LBFGS method. The time is the average reconstruction time per image for $\sigma = 7.65$. The performance of PnP-LBFGS is almost identical to the compared provable PnP methods due to minimizing the same variational form, but with faster convergence.

Scale	$s=2$				$s=3$			
σ	2.55	7.65	12.75	Time (s)	2.55	7.65	12.75	Time (s)
PnP-LBFGS^1	27.89	26.62	25.80	3.19	26.12	25.32	24.68	4.80
PnP-LBFGS^2	27.89	26.62	25.80	9.81	26.12	25.30	24.68	13.15
PnP-PGD	27.44	26.57	25.82	25.99	25.60	25.20	24.63	37.33
PnP-DRSdiff	27.44	26.58	25.82	18.24	25.60	25.19	24.63	32.83
PnP-DRS	27.93	26.61	25.79	15.74	26.13	25.29	24.67	27.00
PnP-$\hat{\alpha}$PGD	27.94	26.62	25.72	4.24	26.11	25.32	24.69	8.78
PnP-FISTA	26.38	26.44	25.79	24.61	24.96	25.15	24.63	33.13
DPIR (iter 10^3)	18.58	26.36	25.74	19.58	17.53	24.96	24.55	19.67
DPIR (iter 24)	27.82	26.60	25.85	0.98	26.06	25.29	24.67	0.97

We observe in Table 4 that the results for PnP-LBFGS are comparable to the other provable PnP methods, with overall faster wall-clock times. In Figure 7 and Figure 8, we are again able to see the difference between the stopping criteria. For the CBSD10 dataset, PnP-LBFGS^1 converges on all images in under 40 iterations, while PnP-LBFGS^2 sometimes requires all 100 iterations, and the other PnP methods take anywhere from 100 to 10^3 iterations to converge. Figure 8 shows again that the convergence of the residuals is significantly faster than the compared PnP methods per iteration. Note that for PnP-LBFGS, PnP-DRS and PnP-$\hat{\alpha}$PGD, we are allowed to choose larger values of the fidelity regularization term λ, leading to better reconstructions in the low noise regime compared to PnP-PGD and PnP-DRSdiff.

As seen in Figure 8c, DPIR does not converge for super-resolution, and we observe an oscillating behavior of the residuals and PSNR. In contrast, PnP-FISTA is able to converge slightly faster than PnP-PGD, but does not converge for some images as seen by the decreasing PSNR for one curve in Figure 7. Both PnP-FISTA and DPIR are able to perform reasonably for higher noise levels of $\sigma = 12.75$, but have more divergence issues for lower noise levels, leading to reduced performance as seen in Table 4. We again observe the gap in performance between DPIR at iteration 10^3 and at iteration 24 as suggested in the original DPIR work. The performance gap between DPIR and provable PnP methods is less apparent for super-resolution as opposed to deblurring, as observed in [32].

4.5. Computational Complexity. While each iteration of PnP-LBFGS has increased complexity, we observed convergence in much fewer iterations. In this section, we outline the computational requirements for the number of neural network N_σ evaluations, denoising steps D_σ, as well as computations of ∇f and $\nabla^2 f$ required per iteration. Note that if a closed form for $\nabla^2 f$ is intractable, computations of (3.5c) can be replaced with Hessian-vector products, available in many deep learning libraries.



(a) PnP-LBFGS[1] (27.87dB) (b) PnP-LBFGS[2] (27.87dB) (c) DPIR (27.58dB)

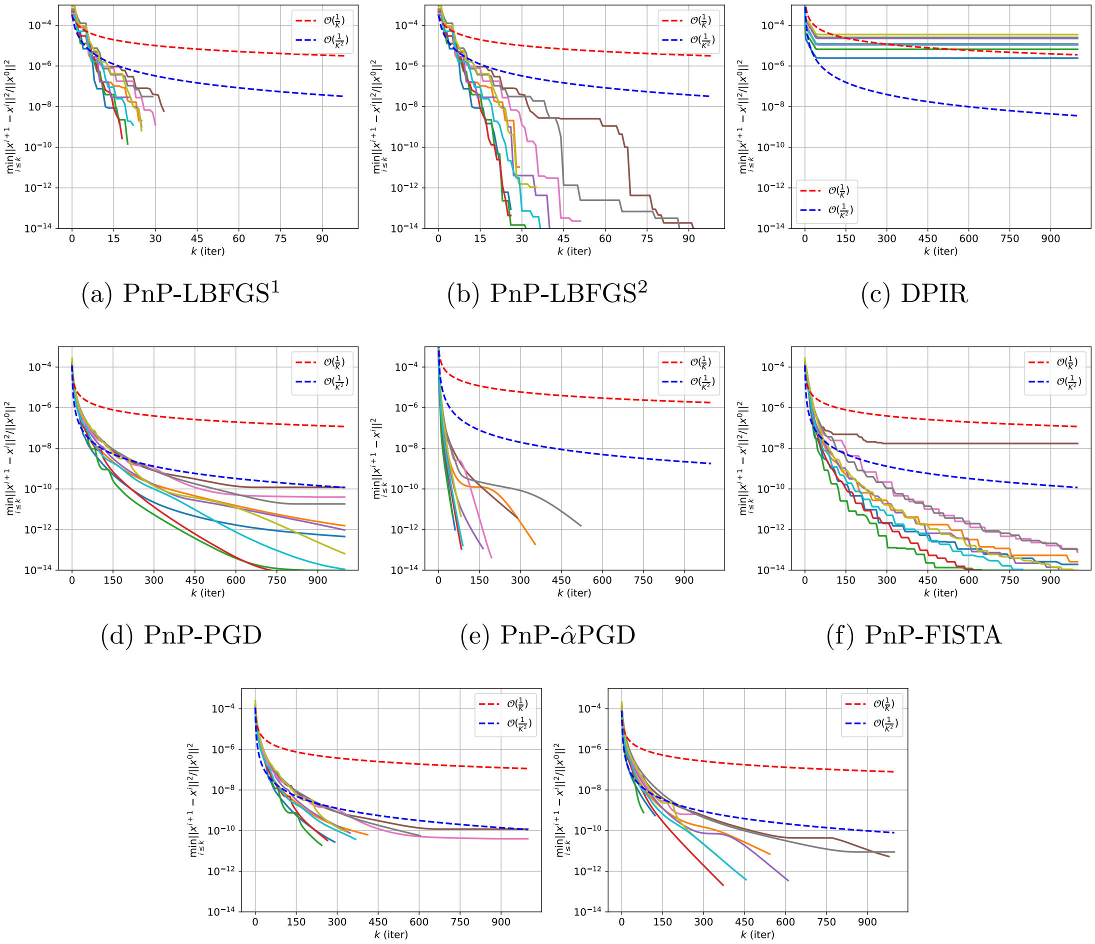
(d) PnP-PGD (27.82dB) (e) PnP-$\hat{\alpha}$PGD (27.86dB) (f) PnP-FISTA (27.71dB)

(g) PnP-DRSdiff (27.69dB) (h) PnP-DRS (27.78dB)

Figure 7: Convergence of the PSNR (dB) of the various curves for super-resolution, with the average dB in brackets. Each curve corresponds to one of the 10 images from the CBSD10 dataset, evaluated with the Gaussian blur kernel with standard deviation $\sigma_{\text{blur}} = 1.2$ and additive noise $\sigma = 7.65$, with scale $s_{sr} = 2$. We observe the convergence of PSNRs in under 40 iterations for PnP-LBFGS[1], much faster than the compared PnP methods.

We can calculate T_γ and R_γ together using one call each of ∇f and D_σ. From (3.5), φ_γ requires ∇f and g_σ, which in turn requires N_σ. $\nabla \varphi_\gamma$ has a closed form, which requires R_γ and an evaluation of $\nabla^2 f$.

Consider a single iteration of PnP-LBFGS. We first compute $\nabla \varphi_\gamma(x^k)$ and $\varphi_\gamma(x^k)$. Computing d^k using L-BFGS does not require any additional evaluations of $D_\sigma, N_\sigma, \nabla f$ or $\nabla^2 f$, as the secants and differences will have been computed in the previous iteration. For each test of w^k, we need to compute a single iteration of φ_γ, which takes one evaluation each of ∇f



(a) PnP-LBFGS[1] (b) PnP-LBFGS[2] (c) DPIR

(d) PnP-PGD (e) PnP-$\hat{\alpha}$PGD (f) PnP-FISTA

(g) PnP-DRSdiff (h) PnP-DRS

Figure 8: Convergence of the residuals $\min_{i\leq k}\|x^{i+1}-x^i\|^2/\|x^0\|^2$ of the various methods for super-resolution. Each curve corresponds to one of the 10 images from the CBSD10 dataset, evaluated with the Gaussian blur kernel with standard deviation $\sigma_{\text{blur}} = 1.2$ and additive noise $\sigma/255 = 7.65$, with scale $s_{sr} = 2$. PnP-LBFGS^2 demonstrates significantly faster residual convergence of the proposed method.

and N_σ. Once a suitable w^k is found, we compute $T_\gamma(w^k)$ and $R_\gamma(w^k)$ together using the last stored $\nabla f(w^k)$, requiring only one additional D_σ operation. For the secant y^k, we require an evaluation of $\nabla\varphi_\gamma(w^k)$, which requires only one additional $\nabla^2 f$ evaluation. This concludes one iteration.

To evaluate the proposed stopping criteria for PnP-LBFGS^1, we are also required to compute $\varphi(x^{k+1})$ from (3.5d). Note we already have $g_\sigma(w^k - \gamma\nabla f(w^k))$ from computing $\varphi_\gamma(w^k)$, and $T_\gamma(w^k) = x^k$, hence we get $\varphi(x^{k+1})$ with no further evaluations needed.

In total, assuming we need T tests for τ_k, the per iteration-cost is

$$\begin{pmatrix} \#N_\sigma \\ \#D_\sigma \\ \#\nabla f \\ \#\nabla^2 f \end{pmatrix}_{\text{PnP-LBFGS}} = \underbrace{\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}}_{\substack{\nabla\varphi_\gamma(x^k), \\ \varphi_\gamma(x^k)}} + T \underbrace{\begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}}_{\text{test } w^k} + \underbrace{\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}}_{\substack{T_\gamma(w^k), \\ R_\gamma(w^k)}} + \underbrace{\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}}_{\nabla\varphi_\gamma(w^k)} = \begin{pmatrix} T+1 \\ 2 \\ T+1 \\ 2 \end{pmatrix}. \tag{4.2}$$

At later iterations, the number of tests is only $T = 1$, since the step-size $\tau = 1$ is accepted almost always. Therefore, later iterations require two of $N_\sigma, D_\sigma, \nabla f$ and $\nabla^2 f$. For comparison, PnP-PGD requires one evaluation each of D_σ and ∇f, and the PnP-DRS methods require one evaluation each of D_σ and prox_f. Note that for these methods to test their stopping criteria by computing φ, they also require one evaluation of g_σ and hence of N_σ [33]. These methods thus have complexity

$$\begin{pmatrix} \#N_\sigma \\ \#D_\sigma \\ \#\nabla f \end{pmatrix}_{\text{PnP-PGD}} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \quad \begin{pmatrix} \#N_\sigma \\ \#D_\sigma \\ \#\operatorname{prox}_f \end{pmatrix}_{\substack{\text{PnP-DRS;} \\ \text{PnP-DRSdiff}}} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.$$

To compute the asymptotic complexity of PnP-LBFGS, suppose the images have dimension d, and that the denoisers have P parameters. From (4.2), we can read off the complexity of computing one iteration given d^k as $\mathcal{O}(d \times P \times T)$, with $\mathcal{O}(d)$ memory requirement to hold the x^k, w^k and intermediate gradients. To compute d^k, the computational complexity of L-BFGS scales linearly with the input dimension and memory length m, and requires us to store m secants and differences. The asymptotic complexity per iteration is thus $\mathcal{O}(d \times P \times T + md)$, where the number of tests T is eventually always 1. The total memory requirement is $\mathcal{O}((m+1) \times d)$, where we store m differences and secants.

A similar complexity analysis can be applied to the PnP-PGD, PnP-DRSdiff and PnP-DRS methods to achieve a per-iteration computational complexity of $\mathcal{O}(d \times P)$ and memory requirement of $\mathcal{O}(d)$. However, these three PnP methods do not come with improved convergence rates under additional smoothness assumptions, and come with residual convergence at a rate $\min_{i \le k} \|x^{i+1} - x^i\| = \mathcal{O}(1/k)$. PnP-LBFGS achieves residual convergence $\min_{i \le k} \|R_{\gamma_i}(x^i)\| = \mathcal{O}(1/k)$ from Theorem 2.11, as well as superlinear convergence under the assumptions of Theorem 2.14. This is summarized in Table 5.

The above complexity analysis shows that the main increase in computational burden for PnP-LBFGS is the requirement of two evaluations of $\nabla^2 f$ at each iteration, as well as at least double the number of neural network evaluations compared to the compared PnP methods. However, assuming only one test for w^k is needed, each iteration only requires one additional evaluation of the denoiser-related networks N_σ, D_σ and fidelity gradient ∇f (or prox_f) to the compared PnP methods. In our experiments, $\nabla^2 f$ has a low computational cost due to the closed form. This allows us to trade roughly 2–3× the per-iteration cost with nearly 10× fewer iterations required as shown in Figures 4 and 8, resulting in fewer total function calls, and thus the 4–5× faster reconstruction times as shown in Tables 3 and 4.

Table 5: Complexity to achieve an ϵ-optimal solution, in terms of the squared residual for PnP-PGD/DRS/DRSdiff, and in terms of the residual $R_{\gamma_i}(x^i)$ for PnP-LBFGS. Under the assumptions of Theorem 2.14 for superlinear convergence, the number of tests is eventually always $T = 1$, and we are able to achieve at least linear speedup.

Complexity	PnP-PGD/DRS/DRSdiff	PnP-LBFGS	PnP-LBFGS superlinear
Computation	$\mathcal{O}(dP\epsilon^{-1})$	$\mathcal{O}\left((dPT + md)\epsilon^{-1}\right)$	$\mathcal{O}\left((dP + md)\log \epsilon\right)$
Memory	$\mathcal{O}(d)$	$\mathcal{O}\left((m + 1)d\right)$	$\mathcal{O}\left((m + 1)d\right)$

5. Conclusion.

In this work, we propose a Plug-and-Play approach to image reconstruction that utilizes descent steps based on the forward-backward envelope. Using the descent formulation, we are able to further incorporate quasi-Newton steps to accelerate convergence. The resulting PnP scheme is provably convergent with a gradient-step assumption on the denoiser by using the Kurdyka-Łojasiewicz property and theoretically achieves superlinear convergence if a Hessian approximation satisfying the Dennis-Moré condition is available. Moreover, properties of the forward-backward envelope allow for additional ways of checking convergence. Our experiments demonstrate that it is able to converge significantly faster in terms of both time and iteration count as well as having highly competitive performance when compared with competing PnP methods with similar convergence guarantees.

For future works, one route is to consider alternative parameterizations of the denoiser D_σ. For example, consider the objective $\varphi = f + \phi_\sigma$ and the task of learning the regularization term ϕ_σ [49, 50]. By enforcing convexity of ϕ_σ through the neural network architecture, such as using input-convex neural networks [1], (weakly-) convex ridge regularizers [25, 26], firm nonexpansiveness [57], or parametric splines [53], results from [67] utilizing convexity such as global sublinear convergence and local linear convergence can be applied. This may also alleviate divergence problems caused when Lipschitz constraints on the denoisers are violated, as sometimes arises using spectral regularization. One restriction of the proposed method lies in the restriction of the regularization parameter, which imposes a bound on the minimum amount of regularization. Future works could look to loosen this restriction, similarly to [31]. In addition, while only simple forward operators such as image deblurring and super-resolution are experimented on in this work, the accelerated convergence rate and model-based interpretation may make this PnP scheme suitable for more complicated forward operators such as CT ray transforms. Future works may explore these practical applications, with a suitably trained "denoiser" for these domains.

Acknowledgements. Hong Ye Tan acknowledges support from GSK.ai and the Masason Foundation. Professor Carola-Bibiane Schönlieb acknowledges support from the Philip Leverhulme Prize, the Royal Society Wolfson Fellowship, the EPSRC advanced career fellowship EP/V029428/1, EPSRC grants EP/S026045/1 and EP/T003553/1, EP/N014588/1, EP/T017961/1, the Wellcome Innovator Awards 215733/Z/19/Z and 221633/Z/20/Z, the European Union Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No. 777826 NoMADS, the Cantab Capital Institute for the Mathematics of Information and the Alan Turing Institute.

REFERENCES

[1] B. Amos, L. Xu, and J. Z. Kolter, *Input convex neural networks*, in International Conference on Machine Learning, PMLR, 2017, pp. 146–155.

[2] S. Arjomand Bigdeli, M. Zwicker, P. Favaro, and M. Jin, *Deep mean-shift priors for image restoration*, Advances in Neural Information Processing Systems, 30 (2017).

[3] L. Armijo, *Minimization of functions having Lipschitz continuous first partial derivatives*, Pacific Journal of mathematics, 16 (1966), pp. 1–3.

[4] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran, *Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Lojasiewicz inequality*, Mathematics of operations research, 35 (2010), pp. 438–457.

[5] H. Attouch, J. Bolte, and B. F. Svaiter, *Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized gauss–seidel methods*, Mathematical Programming, 137 (2013), pp. 91–129.

[6] J.-F. Aujol, C. Dossal, and A. Rondepierre, *Fista is an automatic geometrically optimized algorithm for strongly convex functions*, Mathematical Programming, (2023), pp. 1–43.

[7] H. Bauschke and P. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, CMS Books in Mathematics, Springer New York, 2011.

[8] A. Beck, *First-order methods in optimization*, SIAM, 2017.

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

[10] J. Bolte, S. Sabach, and M. Teboulle, *Proximal alternating linearized minimization for nonconvex and nonsmooth problems*, Mathematical Programming, 146 (2014), pp. 459–494.

[11] A. Buades, B. Coll, and J.-M. Morel, *Non-local means denoising*, Image Processing On Line, 1 (2011), pp. 208–212.

[12] G. T. Buzzard, S. H. Chan, S. Sreehari, and C. A. Bouman, *Plug-and-play unplugged: Optimization-free reconstruction using consensus equilibrium*, SIAM Journal on Imaging Sciences, 11 (2018), pp. 2001–2020.

[13] R. H. Byrd, S. L. Hansen, J. Nocedal, and Y. Singer, *A stochastic quasi-Newton method for large-scale optimization*, SIAM Journal on Optimization, 26 (2016), pp. 1008–1031.

[14] A. Chambolle and C. Dossal, *On the convergence of the iterates of the "fast iterative shrinkage/thresholding algorithm"*, Journal of Optimization theory and Applications, 166 (2015), pp. 968–982.

[15] S. H. Chan, X. Wang, and O. A. Elgendy, *Plug-and-play admm for image restoration: Fixed-point convergence and applications*, IEEE Transactions on Computational Imaging, 3 (2016), pp. 84–98.

[16] R. Cohen, Y. Blau, D. Freedman, and E. Rivlin, *It has potential: Gradient-driven denoisers for convergent solutions to inverse problems*, Advances in Neural Information Processing Systems, 34 (2021), pp. 18152–18164.

[17] R. Cohen, M. Elad, and P. Milanfar, *Regularization by denoising via fixed-point projection (red-pro)*, SIAM Journal on Imaging Sciences, 14 (2021), pp. 1374–1406.

[18] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, *Image denoising by sparse 3-d transform-domain collaborative filtering*, IEEE Transactions on image processing, 16 (2007), pp. 2080–2095.

[19] I. Daubechies, M. Defrise, and C. De Mol, *An iterative thresholding algorithm for linear inverse problems with a sparsity constraint*, Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences, 57 (2004), pp. 1413–1457.

[20] J. E. Dennis and J. J. Moré, *A characterization of superlinear convergence and its application to quasi-Newton methods*, Mathematics of computation, 28 (1974), pp. 549–560.

[21] J. Douglas and H. H. Rachford, *On the numerical solution of heat conduction problems in two and three space variables*, Transactions of the American mathematical Society, 82 (1956), pp. 421–439.

[22] M. Fazlyab, A. Robey, H. Hassani, M. Morari, and G. Pappas, *Efficient and accurate estimation of Lipschitz constants for deep neural networks*, Advances in Neural Information Processing Systems,

32 (2019).
[23] W. Gan, S. Shoushtari, Y. Hu, J. Liu, H. An, and U. S. Kamilov, *Block coordinate plug-and-play methods for blind inverse problems*, arXiv preprint arXiv:2305.12672, (2023).
[24] T. Goldstein, C. Studer, and R. Baraniuk, *A field guide to forward-backward splitting with a fasta implementation*, arXiv preprint arXiv:1411.3406, (2014).
[25] A. Goujon, S. Neumayer, P. Bohra, S. Ducotterd, and M. Unser, *A neural-network-based convex regularizer for image reconstruction*, arXiv preprint arXiv:2211.12461, (2022).
[26] A. Goujon, S. Neumayer, and M. Unser, *Learning weakly convex regularizers for convergent image-reconstruction algorithms*, arXiv preprint arXiv:2308.10542, (2023).
[27] R. Gribonval and M. Nikolova, *A characterization of proximity operators*, Journal of Mathematical Imaging and Vision, 62 (2020), pp. 773–789.
[28] S. Helgason and S. Helgason, *The radon transform*, vol. 2, Springer, 1980.
[29] J. Hertrich, S. Neumayer, and G. Steidl, *Convolutional proximal neural networks and plug-and-play algorithms*, Linear Algebra and its Applications, 631 (2021), pp. 203–234.
[30] W. Huang, K. A. Gallivan, and P.-A. Absil, *A Broyden class of quasi-Newton methods for Riemannian optimization*, SIAM Journal on Optimization, 25 (2015), pp. 1660–1685.
[31] S. Hurault, A. Chambolle, A. Leclaire, and N. Papadakis, *A relaxed proximal gradient descent algorithm for convergent plug-and-play with proximal denoiser*, 2023.
[32] S. Hurault, A. Leclaire, and N. Papadakis, *Gradient step denoiser for convergent plug-and-play*, arXiv preprint arXiv:2110.03220, (2021).
[33] S. Hurault, A. Leclaire, and N. Papadakis, *Proximal denoiser for convergent plug-and-play optimization with nonconvex regularization*, 2022.
[34] Q. Jin, A. Koppel, K. Rajawat, and A. Mokhtari, *Sharpened quasi-Newton methods: Faster superlinear rate and larger local convergence neighborhood*, in International Conference on Machine Learning, PMLR, 2022, pp. 10228–10250.
[35] Q. Jin and A. Mokhtari, *Non-asymptotic superlinear convergence of standard quasi-Newton methods*, Mathematical Programming, 200 (2023), pp. 425–473.
[36] J. Kaipio and E. Somersalo, *Statistical and computational inverse problems*, vol. 160, Springer Science & Business Media, 2006.
[37] U. S. Kamilov, C. A. Bouman, G. T. Buzzard, and B. Wohlberg, *Plug-and-play methods for integrating physical and learned models in computational imaging: Theory, algorithms, and applications*, IEEE Signal Processing Magazine, 40 (2023), pp. 85–97.
[38] U. S. Kamilov, H. Mansour, and B. Wohlberg, *A plug-and-play priors approach for solving nonlinear imaging inverse problems*, IEEE Signal Processing Letters, 24 (2017), pp. 1872–1876.
[39] J. D. Lee, Y. Sun, and M. A. Saunders, *Proximal Newton-type methods for minimizing composite functions*, SIAM Journal on Optimization, 24 (2014), pp. 1420–1443.
[40] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, *Understanding and evaluating blind deconvolution algorithms*, in 2009 IEEE conference on computer vision and pattern recognition, IEEE, 2009, pp. 1964–1971.
[41] D.-H. Li and M. Fukushima, *A modified BFGS method and its global convergence in nonconvex minimization*, Journal of Computational and Applied Mathematics, 129 (2001), pp. 15–35.
[42] D.-H. Li and M. Fukushima, *On the global convergence of the BFGS method for nonconvex unconstrained optimization problems*, SIAM Journal on Optimization, 11 (2001), pp. 1054–1064.
[43] D. C. Liu and J. Nocedal, *On the limited memory BFGS method for large scale optimization*, Mathematical programming, 45 (1989), pp. 503–528.
[44] S. Lunz, O. Öktem, and C.-B. Schönlieb, *Adversarial regularizers in inverse problems*, Advances in neural information processing systems, 31 (2018).
[45] 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*, in Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001, vol. 2, IEEE, 2001, pp. 416–423.
[46] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, *Spectral normalization for generative adversarial networks*, arXiv preprint arXiv:1802.05957, (2018).
[47] A. Mokhtari and A. Ribeiro, *Global convergence of online limited memory BFGS*, The Journal of

Machine Learning Research, 16 (2015), pp. 3151–3181.

[48] J.-J. Moreau, *Proximité et dualité dans un espace hilbertien*, Bulletin de la Société mathématique de France, 93 (1965), pp. 273–299.

[49] S. Mukherjee, S. Dittmer, Z. Shumaylov, S. Lunz, O. Öktem, and C.-B. Schönlieb, *Learned convex regularizers for inverse problems*, 2020.

[50] S. Mukherjee, A. Hauptmann, O. Öktem, M. Pereyra, and C.-B. Schönlieb, *Learned reconstruction methods with convergence guarantees*, arXiv preprint arXiv:2206.05431, (2022).

[51] P. Nair, R. G. Gavaskar, and K. N. Chaudhury, *Fixed-point and objective convergence of plug-and-play algorithms*, IEEE Transactions on Computational Imaging, 7 (2021), pp. 337–348.

[52] S. Neumayer, A. Goujon, P. Bohra, and M. Unser, *Approximation of Lipschitz functions using deep spline neural networks*, SIAM Journal on Mathematics of Data Science, 5 (2023), pp. 306–322.

[53] H. Q. Nguyen, E. Bostan, and M. Unser, *Learning convex regularizers for optimal Bayesian denoising*, IEEE Transactions on Signal Processing, 66 (2017), pp. 1093–1105.

[54] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer, New York, NY, USA, 2e ed., 2006.

[55] P. Ochs and T. Pock, *Adaptive fista for nonconvex optimization*, SIAM Journal on Optimization, 29 (2019), pp. 2482–2503.

[56] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, *Pytorch: An imperative style, high-performance deep learning library*, in Advances in Neural Information Processing Systems 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds., Curran Associates, Inc., 2019, pp. 8024–8035, http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[57] J.-C. Pesquet, A. Repetti, M. Terris, and Y. Wiaux, *Learning maximally monotone operators for image recovery*, SIAM Journal on Imaging Sciences, 14 (2021), pp. 1206–1237.

[58] E. T. Reehorst and P. Schniter, *Regularization by denoising: Clarifications and new interpretations*, IEEE transactions on computational imaging, 5 (2018), pp. 52–67.

[59] R. Rockafellar, *Convex Analysis*, Princeton mathematical series ; 28, Princeton University Press, Princeton, NJ, 1972.

[60] A. Rodomanov and Y. Nesterov, *Greedy quasi-Newton methods with explicit superlinear convergence*, SIAM Journal on Optimization, 31 (2021), pp. 785–811.

[61] A. Rodomanov and Y. Nesterov, *Rates of superlinear convergence for classical quasi-Newton methods*, Mathematical Programming, (2021), pp. 1–32.

[62] D. L. Ruderman, *The statistics of natural images*, Network: Computation in Neural Systems, 5 (1994), pp. 517–548.

[63] L. I. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Physica D: nonlinear phenomena, 60 (1992), pp. 259–268.

[64] E. Ryu, J. Liu, S. Wang, X. Chen, Z. Wang, and W. Yin, *Plug-and-play methods provably converge with properly trained denoisers*, in International Conference on Machine Learning, PMLR, 2019, pp. 5546–5557.

[65] N. N. Schraudolph, J. Yu, and S. Günter, *A stochastic quasi-Newton method for online convex optimization*, in Artificial intelligence and statistics, PMLR, 2007, pp. 436–443.

[66] 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 Transactions on Computational Imaging, 2 (2016), pp. 408–423.

[67] L. Stella, A. Themelis, and P. Patrinos, *Forward–backward quasi-Newton methods for nonsmooth optimization problems*, Computational Optimization and Applications, 67 (2017), pp. 443–487.

[68] Y. Sun, J. Liu, Y. Sun, B. Wohlberg, and U. S. Kamilov, *Async-red: A provably convergent asynchronous block parallel stochastic method using deep denoising priors*, arXiv preprint arXiv:2010.01446, (2020).

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

[70] Y. Sun, Z. Wu, X. Xu, B. Wohlberg, and U. S. Kamilov, *Scalable plug-and-play admm with convergence guarantees*, IEEE Transactions on Computational Imaging, 7 (2021), pp. 849–863.

[71] J. Tang, *Accelerating plug-and-play image reconstruction via multi-stage sketched gradients*, arXiv preprint arXiv:2203.07308, (2022).
[72] E. Tjoa and C. Guan, *A survey on explainable artificial intelligence (xai): Toward medical xai*, IEEE transactions on neural networks and learning systems, 32 (2020), pp. 4793–4813.
[73] A. Vellido, *The importance of interpretability and visualization in machine learning for applications in medicine and health care*, Neural computing and applications, 32 (2020), pp. 18069–18083.
[74] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, *Plug-and-play priors for model based reconstruction*, in 2013 IEEE Global Conference on Signal and Information Processing, IEEE, 2013, pp. 945–948.
[75] X. Wang, S. Ma, D. Goldfarb, and W. Liu, *Stochastic quasi-Newton methods for nonconvex stochastic optimization*, SIAM Journal on Optimization, 27 (2017), pp. 927–956.
[76] F. Wen, L. Chu, P. Liu, and R. C. Qiu, *A survey on nonconvex regularization-based sparse and low-rank recovery in signal processing, statistics, and machine learning*, IEEE Access, 6 (2018), pp. 69883–69906.
[77] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, *Plug-and-play image restoration with deep denoiser prior*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 44 (2021), pp. 6360–6376.
[78] K. Zhang, W. Zuo, S. Gu, and L. Zhang, *Learning deep cnn denoiser prior for image restoration*, in IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3929–3938.