Optimization based machine learning for computational imaging

Optimization plays a central role in modern *machine learning* (**ML**). The most obvious application of optimization in ML is training of deep neural networks, where large-scale algorithms can be used to optimize over billions of parameters and data items. These lecture focuses on a complementary application, where the guiding idea is to adopt concepts from optimization to design ML methods for *computational imaging*.

Computational imaging is a rapidly growing field that seeks to overcome traditional limits of imaging instruments by viewing imaging as an *inverse problem*. Modern computational imaging algorithms achieve impressive results by exploiting complex mathematical models that characterize the physics of the imaging instrument (*forward model*) as well as prior knowledge on the class of desired images (*image prior*). Advances in computational imaging are having transformative effects across a wide range of scientific, engineering, and biomedical applications, including 3D live-cell imaging, structural analysis of complex materials, early diagnosis of neurodegenerative diseases, and improved patient comfort in radiology.

Our focus will be on optimization-based ML methods for solving inverse problems in computational imaging. This remains a very active area of research with many innovations emerging over the past few years. We will focus our exploration on a highly influential class of methods known as *plug-and-play priors* (**PnP**) (see [1] for a recent tutorial). What makes PnP interesting is that it is versatile, mathematically elegant, and state-of-the-art in many imaging applications. PnP is also easily relatable to other ML methods in computational imaging, including *deep unfolding* (**DU**), *deep equilibrium models* (**DEQ**), and diffusion models.

Inverse problems in computational imaging

Let us start by introducing the mathematical notation that we will use throughout these lectures. Our goal is to recover an *unknown* image $\boldsymbol{x} \in \mathbb{R}^n$ from its noisy measurements $\boldsymbol{y} \in \mathbb{R}^m$. We will denote the relationship between the image and its measurements as

$$\boldsymbol{y} = \mathbf{A}(\boldsymbol{x}) + \boldsymbol{e}, \tag{1}$$

where $\mathbf{A}$ is the *measurement operator* that models the physics of the imaging system and $\boldsymbol{e}$ is the *noise*. We will refer to (1) as the *forward model* or the *measurement model*. When the measurement operator is linear, it can be written as a measurement matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$.

Remark. Let us note that the measurement matrices $\mathbf{A}$ are often too large to store or manipulate directly. To see this, consider an inverse problem of recovering a 1MP image (1024×1024 pixels) from its blurry observation. The corresponding matrix $\mathbf{A}$ would have about 10^{12} entries. In practice, we do not need to store $\mathbf{A}$ directly by instead implementing the effect of blur as a convolution.

Examples. Let us consider two examples.

(a) *Image denoising* is an inverse problem that will play a central role in the next chapter. It corresponds to $\mathbf{A} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Denoising seeks to recover two vectors $\boldsymbol{x}$ and $\boldsymbol{e}$ from one vector $\boldsymbol{s} = \boldsymbol{x} + \boldsymbol{e}$, which highlights the necessity of prior information on $\boldsymbol{x}$ and $\boldsymbol{e}$.

(b) In many applications, we have $m < n$, corresponding to an underdetermined system in (1). For example, in image super-resolution, the low-resolution observation $\boldsymbol{y}$ corresponds to a subsampled version of $\boldsymbol{x}$. Since there are ∞ many high-resolution images consistent with $\boldsymbol{y}$, prior on $\boldsymbol{x}$ is essential for the recovery of a meaningful solution.

Denoising networks as image priors

We would like to examine the following denoising problem

$$\boldsymbol{s} = \boldsymbol{x} + \boldsymbol{w}, \quad \boldsymbol{x} \sim p_{\boldsymbol{x}}, \quad \boldsymbol{w} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}), \tag{2}$$

where the goal is to recover $\boldsymbol{x}$ from its noisy observation $\boldsymbol{s}$. The problem (2) assumes that the image $\boldsymbol{x}$ is sampled from some probability distribution $p_{\boldsymbol{x}}$ and $\boldsymbol{w}$ is an *additive white Gaussian noise* (**AWGN**) vector. We are interested in denoising not for denoising, but due to the observation that a deep network pre-trained as a denoiser can give access to the prior $p_{\boldsymbol{x}}$ without requiring explicit knowledge of $p_{\boldsymbol{x}}$ (see Figure 1).

Proximal operator

A key mathematical concept that will enable us to easily deploy denoisers as priors is that of the *proximal operator*. The proximal operator, also known as *proximal mapping*, *proximity operator*, *Moreau proximal operator*, or simply *prox*, was originally introduced by Moreau [2].

Definition 1. The *proximal operator* of $h : \mathbb{R}^n \to (-\infty, \infty]$ is defined as

$$\mathsf{prox}_h(\boldsymbol{z}) = \underset{\boldsymbol{x} \in \mathbb{R}^n}{\mathsf{arg\,min}} \left\{ \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + h(\boldsymbol{x}) \right\}, \quad \boldsymbol{z} \in \mathbb{R}^n.$$

The proximal operator thus maps a vector $\boldsymbol{z} \in \mathbb{R}^n$ into a subset of $\mathbb{R}^n$. The example below show that the proximal operator is *not necessarily unique*. It can be *empty*, *multivalued*, or a *singleton*.

Example. Consider the function

$$h(x) = \begin{cases} 0 & \text{if } x \neq 0 \\ -2 & \text{if } x = 0. \end{cases}$$

The corresponding proximal operator returns the minimizers of the following function

$$\varphi(x) = \frac{1}{2}(x - z)^2 + h(x) = \begin{cases} \frac{1}{2}(x - z)^2 & \text{if } x \neq 0 \\ \frac{1}{2}z^2 - 2 & \text{if } x = 0, \end{cases} \quad \Rightarrow \quad \mathsf{prox}_h(z) = \begin{cases} \{0\} & \text{if } |z| < 2 \\ \{z\} & \text{if } |z| > 2 \\ \{0, z\} & \text{if } |z| = 2. \end{cases}$$

To see this, note that for $z = 0$, the minimizer of φ is clearly $x = 0$. For $z \neq 0$, the minimum of $(x - z)^2/2$ over $\mathbb{R} \setminus \{0\}$ is attained at $x = z \neq 0$ with a minimum value 0. Thus, if $z^2/2 - 2 < 0$, then the unique minimizer of φ is $x = 0$. If $z^2/2 - 2 > 0$, then the unique minimizer of φ is $x = z$. Finally, when $z^2/2 - 2 = 0$, then 0 and z are the two minimizers of p.

The next result establishes conditions for the existence of prox.

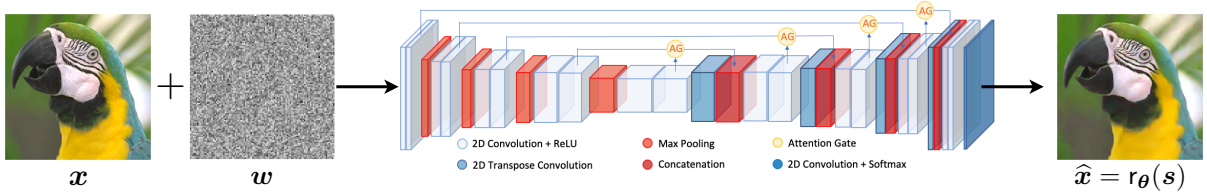


Figure 1: Priors in modern PnP methods are typically obtained by training a deep network as a denoiser.

Proposition 1. Let $h : \mathbb{R}^n \to (-\infty, \infty]$ be a proper and closed function. Assume that the function

$$\varphi(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + h(\boldsymbol{x}),$$

is *coercive* for any $\boldsymbol{z} \in \mathbb{R}^n$. Then, $\mathsf{prox}_h(\boldsymbol{z})$ is nonempty for any $\boldsymbol{z} \in \mathbb{R}^n$.

Remark. As a reminder, we say that φ is *corcive* if

$$\lim_{\|\boldsymbol{x}\|_2 \to \infty} \varphi(\boldsymbol{x}) = +\infty.$$

This means that

$$\forall \varphi_0 \in \mathbb{R}, \exists r > 0 \quad \text{s.t.} \quad \varphi(\boldsymbol{x}) > \varphi_0 \quad \text{whenever} \quad \|\boldsymbol{x}\| > r.$$

Thus, a coercive function φ becomes $+\infty$ for any path for which $\|\boldsymbol{x}\|_2$ becomes $+\infty$. A classical result in optimization states that proper, closed, and coercive functions always have a minimizer.

Proof. To prove Thereom 1, note that for any $\boldsymbol{z} \in \mathbb{R}^n$, φ is closed as a sum of two closed functions. Since φ is coercive, we know that $\mathsf{prox}_f(\boldsymbol{z})$, which consists of minimizers of φ, is nonempty. ■

Proximal operators are commonly used for proper, closed, and convex functions $f \in \Gamma^0(\mathbb{R}^n)$. One reason is that the proximal operator of such functions always exists and is unique.

Proposition 2. Let $h \in \Gamma^0(\mathbb{R}^n)$, then prox_h is a singleton for any $\boldsymbol{z} \in \mathbb{R}^n$.

Remark. We will use the standard notation $\Gamma^0(\mathbb{R}^n)$ to denote the class of proper, closed, and convex functions. When the proximal operator is a singleton, we drop the set notation $\{\cdot\}$ from its output.

Proof. The function

$$\varphi(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + h(\boldsymbol{x}).$$

is closed and strongly convex, since it is a sum of a closed strongly convex quadratic and closed convex h. Since h is proper, we know that φ is also proper. Since a strongly convex function always has a unique minimizer, we know that there exists a unique minimizer of φ. ■

Example: Quadratic and indicator functions

(a) The proximal operator of a quadratic function has a closed form expression

$$h(\boldsymbol{x}) = \frac{\tau}{2}\|\boldsymbol{x}\|_2^2 \quad \Rightarrow \quad \mathsf{prox}_h(\boldsymbol{z}) = \operatorname*{arg\,min}_{\boldsymbol{x} \in \mathbb{R}^n} \left\{ \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + \frac{\tau}{2}\|\boldsymbol{x}\|_2^2 \right\} = \frac{\boldsymbol{z}}{1+\tau}.$$

(b) The proximal operator of an indicator function of a nonempty, closed, and convex set $\mathcal{X} \subseteq \mathbb{R}^n$ is the *projection*

$$\begin{aligned} h(\boldsymbol{x}) = \mathbb{1}_{\mathcal{X}}(\boldsymbol{x}) \quad \Rightarrow \quad \mathsf{prox}_h(\boldsymbol{z}) &= \operatorname*{arg\,min}_{\boldsymbol{x} \in \mathbb{R}^n} \left\{ \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + \mathbb{1}_{\mathcal{X}}(\boldsymbol{x}) \right\} \\ &= \operatorname*{arg\,min}_{\boldsymbol{x} \in \mathcal{X}} \left\{ \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 \right\} = \mathsf{proj}_{\mathcal{X}}(\boldsymbol{z}). \end{aligned}$$

The proximal operator has a number of important properties related to the fact that it solves an optimization problem. In particular, it can be fully characterized using a subdifferential.

Proposition 3. For $h \in \Gamma^0(\mathbb{R}^n)$, the following relationships are equivalent for any $\boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^n$

$$\boldsymbol{x} = \mathsf{prox}_h(\boldsymbol{z}) \quad \Leftrightarrow \quad (\boldsymbol{z} - \boldsymbol{x}) \in \partial h(\boldsymbol{x}) \quad \Leftrightarrow \quad (\boldsymbol{z} - \boldsymbol{x})^{\mathsf{T}}(\boldsymbol{y} - \boldsymbol{x}) \leq h(\boldsymbol{y}) - h(\boldsymbol{x}), \quad \forall \boldsymbol{y} \in \mathbb{R}^n.$$

Remark: Projection theorem. Classical *projection theorem* is the special case of Proposition 3. Let $h(\boldsymbol{x}) = \mathbb{1}_{\mathcal{X}}(\boldsymbol{x})$, where $\mathcal{X} \subseteq \mathbb{R}^n$ is nonempty, closed, and convex. Then, for any $\boldsymbol{z} \in \mathbb{R}^n$, we have

$$\boldsymbol{x} = \mathsf{prox}_h(\boldsymbol{z}) = \mathsf{proj}_{\mathcal{X}}(\boldsymbol{z}) \quad \Leftrightarrow \quad (\boldsymbol{z} - \mathsf{proj}_{\mathcal{X}}(\boldsymbol{z}))^{\mathsf{T}}(\boldsymbol{y} - \mathsf{proj}_{\mathcal{X}}(\boldsymbol{z})) \leq 0, \quad \forall \boldsymbol{y} \in \mathcal{X}.$$

Proof. We can apply the first-order optimality condition to the definition of the proximal, which leads to

$$\boldsymbol{0} \in \boldsymbol{x} - \boldsymbol{z} + \partial h(\boldsymbol{x}) \quad \Leftrightarrow \quad (\boldsymbol{z} - \boldsymbol{x}) \in \partial h(\boldsymbol{x}).$$

Additionally, from the definition of the subgradient, we have for any $\boldsymbol{g}(\boldsymbol{x}) \in \partial h(\boldsymbol{x})$

$$h(\boldsymbol{y}) \geq h(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})^{\mathsf{T}}(\boldsymbol{y} - \boldsymbol{x}),$$

which directly leads to the last equivalence. ■

An important consequence of Theorem 3 is that a *fixed point* of the proximal operator minimizes h.

Corollary 1. Let $h \in \Gamma^0(\mathbb{R}^n)$ and $\boldsymbol{x} \in \mathbb{R}^n$, then

$$\boldsymbol{x} \text{ is a minimizer of } h \quad \Leftrightarrow \quad \boldsymbol{x} = \mathsf{prox}_h(\boldsymbol{x}).$$

Proof. We know that $\boldsymbol{x}$ is a minimizer of h if and only if $\boldsymbol{0} \in \partial h(\boldsymbol{x})$. Thus, from theorem above we obtain the equivalence with $\boldsymbol{x} = \mathsf{prox}_h(\boldsymbol{x})$. ■

Finally, one can use Theorem 3 to establish the firm nonexpansiveness of the proximal operator.

Proposition 4. Let $h \in \Gamma^0(\mathbb{R}^n)$, then the proximal operator is *firmly nonexpansive*, which means that

$$\|\mathsf{prox}_h(\boldsymbol{x}) - \mathsf{prox}_h(\boldsymbol{y})\|_2^2 \leq (\mathsf{prox}_h(\boldsymbol{x}) - \mathsf{prox}_h(\boldsymbol{y}))^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y}), \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n.$$

By using Cauchy-Schwarz inequality, we can further show that the proximal operator is *nonexpansive*

$$\|\mathsf{prox}_h(\boldsymbol{x}) - \mathsf{prox}_h(\boldsymbol{y})\|_2 \leq \|\boldsymbol{x} - \boldsymbol{y}\|_2, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n.$$

Proof. Let $\boldsymbol{u} = \mathsf{prox}_h(\boldsymbol{x})$ and $\boldsymbol{v} = \mathsf{prox}_h(\boldsymbol{y})$. Then, we have that

$$\begin{cases} (\boldsymbol{x} - \boldsymbol{u}) \in \partial h(\boldsymbol{u}) \\ (\boldsymbol{y} - \boldsymbol{v}) \in \partial h(\boldsymbol{v}) \end{cases} \quad \Rightarrow \quad (\boldsymbol{x} - \boldsymbol{u} - \boldsymbol{y} + \boldsymbol{v})^{\mathsf{T}}(\boldsymbol{u} - \boldsymbol{v}) \geq 0 \quad \Rightarrow \quad (\boldsymbol{x} - \boldsymbol{y})^{\mathsf{T}}(\boldsymbol{u} - \boldsymbol{v}) \geq \|\boldsymbol{u} - \boldsymbol{v}\|_2^2,$$

where in the first implication we used the monotonicity of the subdifferential. ■

Characterization of proximal operators

Given a function h, one can derive the value of the proximal operator $\mathsf{R} = \mathsf{prox}_h$. In this section, we are interested in the reverse question: given an operator R, what conditions would make it a proximal operator? We will start with the following definition.

Definition 2. An operator $\mathsf{R} : \mathcal{X} \to \mathbb{R}^n$ is a proximal operator of h if $\mathsf{R}(\boldsymbol{z}) \in \mathsf{prox}_h(\boldsymbol{x})$ for each $\boldsymbol{x} \in \mathcal{X}$.

Proximal operators for functions in $\Gamma^0(\mathbb{R}^n)$ were characterized by Moreau in his original paper [2].

Proposition 5. An operator $\mathsf{R} : \mathbb{R}^n \to \mathbb{R}^n$ is the proximal operator of $h \in \Gamma^0(\mathbb{R}^n)$ if and only if the following two conditions are true:

(a) there exists a convex and closed function ψ such that: $\mathsf{R}(\boldsymbol{x}) \in \partial\psi(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^n$;

(b) R is nonexpansive: $\|\mathsf{R}(\boldsymbol{x}) - \mathsf{R}(\boldsymbol{z})\|_2 \leq \|\boldsymbol{x} - \boldsymbol{z}\|_2$ for all $\boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^n$.

Proof. See Corollary 10.c in [2]. ■

The original result by Moreau was extended to nonconvex functions by Gribonval and Nikolova [3].

Proposition 6. Consider $\mathsf{R} : \mathcal{X} \to \mathbb{R}^n$, where $\mathcal{X} \subset \mathbb{R}^n$ is nonempty. We have that

There exists $h : \mathbb{R}^n \to (-\infty, +\infty]$ such that $\mathsf{R}(\boldsymbol{x}) \in \mathsf{prox}_h(\boldsymbol{x})$, $\forall \boldsymbol{x} \in \mathcal{X}$
$\Leftrightarrow$ There exists convex and closed $\psi : \mathbb{R}^n \to (-\infty, +\infty]$ such that $\mathsf{R}(\boldsymbol{x}) \in \partial\psi(\boldsymbol{x})$, $\forall \boldsymbol{x} \in \mathcal{X}$.

Proof. See Theorem 3 in [3]. ■

Remark. The analysis in Proposition 6 further implies that h and ψ can be chosen such that

$$\psi(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}}\mathsf{R}(\boldsymbol{x}) - \frac{1}{2}\|\mathsf{R}(\boldsymbol{x})\|_2^2 - h(\mathsf{R}(\boldsymbol{x})), \quad \forall \boldsymbol{x} \in \mathcal{X}.$$

The results [3] can be used to recover classical characterization of certain proximal operators.

Proposition 7. Consider $\mathsf{R} : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X} \subset \mathbb{R}$ is nonempty. We have that

R is nondecreasing $\Leftrightarrow$ There exists $h : \mathbb{R} \to (-\infty, +\infty]$ such that $\mathsf{R}(\boldsymbol{x}) \in \mathsf{prox}_h(\boldsymbol{x})$, $\forall \boldsymbol{x} \in \mathcal{X}$.

Proof. See Corollary 7 in [3]. ■

Example. Consider the classical soft-thresholding function

$$\mathsf{T}_\mu(z) = \mathsf{sign}(z) \cdot \mathsf{max}(|z| - \mu, 0), \quad z \in \mathbb{R}.$$

Since T_μ is nondecreasing, it is a proximal operator. In fact, it is known that it the soft-thresholding function is the proximal operator of $h(x) = \mu|x|$.

Moreau smoothing

We will next discuss the concept of Moreau envelope for proper, closed, and convex functions. This will give us an elegant interpretation of the proximal operator of h as a gradient-descent step of some function h_μ.

Definition 3. The Moreau envelope of $h \in \Gamma^0(\mathbb{R}^n)$ is defined as

$$h_\mu(\boldsymbol{z}) = \inf_{\boldsymbol{x}\in\mathbb{R}^n} \left\{ \frac{1}{2\mu}\|\boldsymbol{x}-\boldsymbol{z}\|_2^2 + h(\boldsymbol{x}) \right\}, \quad \boldsymbol{z} \in \mathbb{R}^n.$$

Example (Huber function). Let $h(z) = |z|$. Then, its Moreau envelope is just the Huber function

$$h_\mu(z) = \inf_{x} \left\{ \frac{1}{2\mu}(x-z)^2 + |x| \right\} = \begin{cases} \frac{1}{2\mu}z^2 & \text{for } |z| \leq \mu \\ |z| - \frac{\mu}{2} & \text{for } |z| > \mu \end{cases}.$$

Proposition 8. The Moreau envelope h_μ of $h \in \Gamma^0(\mathbb{R}^n)$ is convex and has a $(1/\mu)$-Lipschitz gradient

$$\nabla h_\mu(\boldsymbol{z}) = \frac{1}{\mu}\left(\boldsymbol{z} - \mathsf{prox}_{\mu h}(\boldsymbol{z})\right), \quad \boldsymbol{z} \in \mathbb{R}^n.$$

Remark. Note that Proposition 8 implies that

$$\mathsf{prox}_{\mu h}(\boldsymbol{x}) = \boldsymbol{x} - \mu\nabla h_\mu(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^n,$$

which allows us to conclude that the proximal operator is computing a gradient-descent step on h_μ.

Proof. We first show that h_μ is convex. Consider

$$\varphi(\boldsymbol{x},\boldsymbol{z}) = \frac{1}{2\mu}\|\boldsymbol{x}-\boldsymbol{z}\|_2^2 + h(\boldsymbol{x}),$$

which is convex over $(\boldsymbol{x},\boldsymbol{z})$. Then, for any $0 \leq \theta \leq 1$ and any $(\boldsymbol{x}_1,\boldsymbol{z}_1),(\boldsymbol{x}_2,\boldsymbol{z}_2)$ in $\mathbb{R}^{2n}$, we have

$$\begin{aligned} h_\mu(\theta\boldsymbol{z}_1 + (1-\theta)\boldsymbol{z}_2) &\leq \varphi(\theta\boldsymbol{x}_1 + (1-\theta)\boldsymbol{x}_2, \theta\boldsymbol{z}_1 + (1-\theta)\boldsymbol{z}_2) \\ &\leq \theta\varphi(\boldsymbol{x}_1,\boldsymbol{z}_1) + (1-\theta)\varphi(\boldsymbol{x}_2,\boldsymbol{z}_2), \end{aligned}$$

where we used the convexity of φ. Since this inequality holds everywhere, we have

$$h_\mu(\theta\boldsymbol{z}_1 + (1-\theta)\boldsymbol{z}_2) \leq \theta h_\mu(\boldsymbol{z}_1) + (1-\theta)h_\mu(\boldsymbol{z}_2),$$

with

$$h_\mu(\boldsymbol{z}_1) = \inf_{\boldsymbol{x}_1} \varphi(\boldsymbol{x}_1,\boldsymbol{z}_1) \quad \text{and} \quad h_\mu(\boldsymbol{z}_2) = \inf_{\boldsymbol{x}_2} \varphi(\boldsymbol{x}_2,\boldsymbol{z}_2).$$

To show the differentiability, note that

$$\begin{aligned} h_\mu(\boldsymbol{z}) &= \frac{1}{2\mu}\|\boldsymbol{z}\|_2^2 - \frac{1}{\mu}\sup_{\boldsymbol{x}\in\mathbb{R}^n}\left\{\boldsymbol{z}^\mathsf{T}\boldsymbol{x} - \mu h(\boldsymbol{x}) - \frac{1}{2}\|\boldsymbol{x}\|_2^2\right\} \\ &= \frac{1}{2\mu}\|\boldsymbol{z}\|_2^2 - \frac{1}{\mu}\phi^*(\boldsymbol{z}) \quad \text{with} \quad \phi^*(\boldsymbol{z}) \coloneqq \frac{1}{2}\|\boldsymbol{z}\|_2^2 + \mu h(\boldsymbol{z}), \end{aligned}$$

where ϕ^* denotes the conjugate function of ϕ. The function ϕ is closed and 1-strongly convex. Hence, we know that ϕ^* is defined for all $\boldsymbol{z} \in \mathbb{R}^n$ and that the gradient of the conjugate function $\phi^*(\boldsymbol{z})$ is equal to the optimal $\boldsymbol{x}^\star$ at which $\phi^*(\boldsymbol{x})$ is achieved

$$\nabla \phi^*(\boldsymbol{z}) = \operatorname*{arg\,max}_{\boldsymbol{x} \in \mathbb{R}^n} \left\{ \boldsymbol{z}^\mathsf{T} \boldsymbol{x} - \mu h(\boldsymbol{x}) - \frac{1}{2}\|\boldsymbol{x}\|_2^2 \right\} = \mathsf{prox}_{\mu h}(\boldsymbol{z}).$$

Hence, we conclude that

$$\frac{1}{\mu}\left(\boldsymbol{z} - \mathsf{prox}_{\mu h}(\boldsymbol{z})\right) = \nabla h_\mu(\boldsymbol{z}).$$

Note that since the proximal operator is firmly nonexpansive, $\mu \nabla h_\mu$ is also firmly nonexpansive, which implies that it is $(1/\mu)$- Lipschitz continuous. ■

We next show that the Moreau envelope can serve as a smooth approximation to a nonsmooth function.

Proposition 9. Consider $h(\boldsymbol{z})$, its Moreau envelope $h_\mu(\boldsymbol{z})$ for $\mu > 0$, and any $\boldsymbol{g}(\boldsymbol{z}) \in \partial h(\boldsymbol{z})$. Then,

$$0 \leq h(\boldsymbol{z}) - h_\mu(\boldsymbol{z}) \leq \frac{\mu}{2}\|\boldsymbol{g}(\boldsymbol{z})\|_2^2.$$

Proof. First note that we have for any $\boldsymbol{z} \in \mathbb{R}^n$

$$h_\mu(\boldsymbol{z}) = \inf_{\boldsymbol{x}} \left\{ \frac{1}{2\mu}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + h(\boldsymbol{x}) \right\} \leq h(\boldsymbol{z}),$$

which is due to the fact that $\boldsymbol{x} = \boldsymbol{z}$ is potentially suboptimal. We additionally have for any $\boldsymbol{g}(\boldsymbol{z}) \in \partial h(\boldsymbol{z})$

$$\begin{aligned} h_\mu(\boldsymbol{z}) - h(\boldsymbol{z}) &= \inf_{\boldsymbol{x}} \left\{ h(\boldsymbol{x}) - h(\boldsymbol{z}) + \frac{1}{2\mu}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 \right\} \\ &\geq \inf_{\boldsymbol{x}} \left\{ \boldsymbol{g}(\boldsymbol{z})^\mathsf{T}(\boldsymbol{x} - \boldsymbol{z}) + \frac{1}{2\mu}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 | \right\} \\ &= \inf_{\boldsymbol{x}} \left\{ \frac{1}{2\mu}\|\boldsymbol{x} - (\boldsymbol{z} - \mu \boldsymbol{g}(\boldsymbol{z}))\|_2^2 - \frac{\mu}{2}\|\boldsymbol{g}(\boldsymbol{z})\|_2^2 \right\} = -\frac{\mu}{2}\|\boldsymbol{g}(\boldsymbol{z})\|_2^2. \end{aligned}$$

This directly leads to the conclusion. ■

Bayesian estimation

We are now equipped with tools to establish a relationship between an estimation probelm (2) and the proximal operator. We will adopt the Bayesian perspective by designing an operator $\mathsf{R}(\boldsymbol{s})$ for (2) that minimizes

$$\mathbb{E}_{\boldsymbol{s},\boldsymbol{x}} \left[\mathcal{L}(\boldsymbol{x}, \mathsf{R}(\boldsymbol{s})) \right], \tag{3}$$

where $\mathcal{L}(\boldsymbol{x}, \widehat{\boldsymbol{x}})$ is some loss function and $(\boldsymbol{x}, \boldsymbol{s})$ is drawn from the joint distribution $p_{\boldsymbol{x},\boldsymbol{s}}(\boldsymbol{x}, \boldsymbol{s}) = p_{\boldsymbol{s}|\boldsymbol{x}}(\boldsymbol{s}|\boldsymbol{x})\, p_{\boldsymbol{x}}(\boldsymbol{x})$. Note that the conditional distribution of $\boldsymbol{s}$ given $\boldsymbol{x}$ is simply the Gaussian probability density function

$$p_{\boldsymbol{s}|\boldsymbol{x}}(\boldsymbol{s}|\boldsymbol{x}) = \mathcal{G}_\sigma(\boldsymbol{s} - \boldsymbol{x}) = \frac{1}{\left(\sqrt{2\pi\sigma^2}\right)^n} \exp\left(-\frac{\|\boldsymbol{s} - \boldsymbol{x}\|_2^2}{2\sigma^2} \right),$$

where we introduce the notation $\mathcal{G}_\sigma$ for the Gaussian density with the standard deviation $\sigma > 0$.

Example. The *maximum a posteriori* (**MAP**) estimator is often been used to justify various inverse problem solvers. MAP can be seen as minimizing (3) with the pseudo-cost $\mathcal{L}(\boldsymbol{x}, \widehat{\boldsymbol{x}}) = \delta(\boldsymbol{x} - \widehat{\boldsymbol{x}})$. A MAP denoiser simply selects a vector that maximizes $p(\boldsymbol{x}|\boldsymbol{s}) \propto p(\boldsymbol{x}|\boldsymbol{s})p_{\boldsymbol{x}}(\boldsymbol{x})$

$$\mathsf{R}_{\mathsf{map}}(\boldsymbol{s}) \in \operatorname*{arg\,min}_{\boldsymbol{x}\in\mathbb{R}^n} p(\boldsymbol{x}|\boldsymbol{s}) = \operatorname*{arg\,min}_{\boldsymbol{x}\in\mathbb{R}^n} \left\{ \frac{1}{2\sigma^2}\|\boldsymbol{x} - \boldsymbol{s}\|_2^2 - \log p_{\boldsymbol{x}}(\boldsymbol{x}) \right\}.$$

Therefore, a MAP denoiser can be directly expressed as the proximal operator

$$\mathsf{R}_{\mathsf{map}}(\boldsymbol{s}) \in \mathsf{prox}_{\mu h_{\mathsf{map}}}(\boldsymbol{s}) \quad \text{with } \mu := \sigma^2 \text{ and } h_{\mathsf{map}}(\boldsymbol{x}) := -\log p_{\boldsymbol{x}}(\boldsymbol{x}).$$

Example. The *minimum mean squared error* (**MMSE**) estimator seeks to minimize (3) with the cost $\mathcal{L}(\boldsymbol{x}, \widehat{\boldsymbol{x}}) = \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|_2^2$ and the solution is the conditional expectation

$$\mathsf{R}_{\mathsf{mmse}}(\boldsymbol{s}) = \mathbb{E}[\boldsymbol{x}|\boldsymbol{s}] = \int_{\mathbb{R}^n} \boldsymbol{x}\, p(\boldsymbol{x}|\boldsymbol{s})\, \mathrm{d}\boldsymbol{x} = \frac{1}{p(\boldsymbol{s})} \int_{\mathbb{R}^n} \boldsymbol{x}\, \mathcal{G}_\sigma(\boldsymbol{x} - \boldsymbol{s})\, p_{\boldsymbol{x}}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}.$$

Proposition 10 (Tweedie formula). Let $\mathsf{R}_{\mathsf{mmse}}(\boldsymbol{s}) = \mathbb{E}[\boldsymbol{x}|\boldsymbol{s}]$ for problem (2), then we have

$$\mathsf{R}_{\mathsf{mmse}}(\boldsymbol{s}) = \boldsymbol{s} + \sigma^2 \nabla \log p(\boldsymbol{s}), \quad \boldsymbol{s} \in \mathbb{R}^n.$$

Proof. First note that gradient of the Gaussian density is given by

$$\nabla_{\boldsymbol{s}} \mathcal{G}_\sigma(\boldsymbol{s} - \boldsymbol{x}) = \frac{1}{\sigma^2}(\boldsymbol{x} - \boldsymbol{s})\mathcal{G}_\sigma(\boldsymbol{s} - \boldsymbol{x}).$$

Now consider the marginal distribution and its gradient

$$\begin{aligned} p(\boldsymbol{s}) &= \int p(\boldsymbol{x}, \boldsymbol{s})\, \mathrm{d}\boldsymbol{x} = \int \mathcal{G}_\sigma(\boldsymbol{s} - \boldsymbol{s}) p_{\boldsymbol{x}}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \\ \Rightarrow \quad \nabla p(\boldsymbol{s}) &= \frac{1}{\sigma^2} \int (\boldsymbol{x} - \boldsymbol{s}) \mathcal{G}_\sigma(\boldsymbol{s} - \boldsymbol{x}) p_{\boldsymbol{x}}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} = \frac{1}{\sigma^2} \left[\int \boldsymbol{x}\, p(\boldsymbol{x}, \boldsymbol{s})\, \mathrm{d}\boldsymbol{x} - \boldsymbol{s}\, p(\boldsymbol{s}) \right], \end{aligned}$$

where we used the gradient of the Gaussian density and $p(\boldsymbol{x}, \boldsymbol{s}) = \mathcal{G}_\sigma(\boldsymbol{s} - \boldsymbol{x}) p_{\boldsymbol{x}}(\boldsymbol{x})$. Dividing both sides by $p(\boldsymbol{s})$ and using the definition of conditional mean, we obtain

$$\nabla \log p(\boldsymbol{s}) = \frac{\nabla p(\boldsymbol{s})}{p(\boldsymbol{s})} = \frac{1}{\sigma^2} \left(\mathbb{E}[\boldsymbol{x}|\boldsymbol{s}] - \boldsymbol{s} \right),$$

which establishes the desired result. ■

MAP and MMSE denoisers are generally not equivalent. However, they are when the prior is Gaussian.

Example. Consider the Gaussian prior $p_{\boldsymbol{x}}(\boldsymbol{x}) = \mathcal{G}_\nu(\boldsymbol{x})$ where $\nu > 0$. Then, we have that

$$\mathsf{R}_{\mathsf{map}}(\boldsymbol{s}) = \mathsf{R}_{\mathsf{mmse}}(\boldsymbol{s}) = \left(\frac{\nu^2}{\nu^2+\sigma^2}\right)\boldsymbol{s}.$$

Note how in the Gaussian case, both estimators correspond to the same *linear* function.

Proof. Let $p_{\boldsymbol{x}}(\boldsymbol{x}) = \mathcal{G}_\nu(\boldsymbol{x})$ where $\nu > 0$. For MAP, we have that

$$\mathsf{R}_{\mathsf{map}}(\boldsymbol{s}) = \operatorname*{arg\,min}_{\boldsymbol{x}} \left\{\frac{1}{2\sigma^2}\|\boldsymbol{x}-\boldsymbol{s}\|_2^2 + \frac{1}{2\nu^2}\|\boldsymbol{x}\|_2^2\right\} = \left(\frac{\nu^2}{\nu^2+\sigma^2}\right)\boldsymbol{s}.$$

For MMSE, we have that

$$p(\boldsymbol{s}) = \int \mathcal{G}_\sigma(\boldsymbol{s}-\boldsymbol{x})\,\mathcal{G}_\nu(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \mathcal{G}_\alpha(\boldsymbol{s}) \quad \Rightarrow \quad \log p(\boldsymbol{s}) = -\frac{1}{\alpha^2}\|\boldsymbol{s}\|_2^2,$$

where $\alpha^2 \coloneqq \sigma^2 + \nu^2$. Using the Tweedie formula, we obtain

$$\mathsf{R}_{\mathsf{mmse}}(\boldsymbol{s}) = \boldsymbol{s} - \left(\frac{\sigma^2}{\nu^2+\sigma^2}\right)\boldsymbol{s} = \left(\frac{\nu^2}{\nu^2+\sigma^2}\right)\boldsymbol{s}.$$

■

The following result is due to Gribonval [4] and shows that the MMSE denoiser is a proximal operator.

Proposition 11. For a non-degenerate prior $p_{\boldsymbol{x}}$, there exists a function h_{mmse} such that

$$\mathsf{R}_{\mathsf{mmse}}(\boldsymbol{s}) = \mathsf{prox}_{h_{\mathsf{mmse}}}(\boldsymbol{s}), \quad \forall \boldsymbol{s} \in \mathbb{R}^n.$$

The function h_{mmse} is given by

$$h_{\mathsf{mmse}}(\boldsymbol{x}) = \begin{cases} -\frac{1}{2}\|\boldsymbol{x} - \mathsf{R}_{\mathsf{mmse}}^{-1}(\boldsymbol{x})\|_2^2 - \sigma^2 \log p_{\boldsymbol{s}}(\mathsf{R}_{\mathsf{mmse}}^{-1}(\boldsymbol{x})) & \text{for } \boldsymbol{x} \in \mathcal{X} \\ +\infty & \text{for } \boldsymbol{x} \notin \mathcal{X} \end{cases},$$

where $\mathcal{X} \coloneqq \mathsf{Im}(\mathsf{R}_{\mathsf{mmse}})$, $\mathsf{R}_{\mathsf{mmse}}^{-1} : \mathcal{X} \to \mathbb{R}^n$ is the inverse mapping, which is well defined and smooth over $\mathcal{X}$, and $p_{\boldsymbol{s}}$ is the probability distribution of the AWGN corrupted observation in (2).

Proof. See Theorem 1 in [4]. ■

Remark. Proposition 11 states that there exists a function h_{mmse} such that the MMSE denoiser is its proximal operator. The underlying analysis additionally shows that h_{mmse} is C^∞ over $\mathsf{Im}(\mathsf{R}_{\mathsf{mmse}})$. One interesting conclusion from this results is that one can in principle misinterpret a MMSE denoiser as a MAP denoiser corresponding to a prior probability $q(\boldsymbol{x}) \propto \exp(-h_{\mathsf{mmse}}(\boldsymbol{x}))$.

The next result provides necessary and sufficient conditions on the convexity and separability of the implicit regularizer h specified via the MMSE estimator.

Proposition 12. Let $p_{\boldsymbol{x}}$ be a non-degenerate prior and h be the regularizer associated with R_{mmse} via Proposition 11. The following statements are true:

(a) h is convex if and only if $p_{\boldsymbol{s}} = (\mathcal{G}_\sigma * p_{\boldsymbol{x}})$ is log-concave.

(b) h is additively separable if and only if $p_{\boldsymbol{s}}$ is multiplicatively separable.

Proof. See Lemma 2 in [5]. ■

Remark. Proposition 12 uses the following notions of separability.

(a) $\mathsf{R} : \mathbb{R}^n \to \mathbb{R}^n$ is separable if there exists a set of functions $\mathsf{R}_1, \ldots, \mathsf{R}_n : \mathbb{R} \to \mathbb{R}$ such that $\mathsf{R}(\boldsymbol{x}) = (\mathsf{R}_1(x_1), \cdots, \mathsf{R}_n(x_n))$, for all $\boldsymbol{x} \in \mathbb{R}^n$.

(b) $h : \mathbb{R}^n \to \mathbb{R}$ is additively separable if there exists a set of functions $h_1, \ldots, h_n : \mathbb{R} \to \mathbb{R}$ such that $h(\boldsymbol{x}) = h_1(x_1) + \cdots + h_n(x_n)$ for all $\boldsymbol{x} \in \mathbb{R}^n$.

(c) $h : \mathbb{R}^n \to \mathbb{R}$ is multiplicatively separable if there exists a set of functions $h_1, \ldots, h_n : \mathbb{R} \to \mathbb{R}$ such that $h(\boldsymbol{x}) = h_1(x_1) \cdots h_n(x_n)$ for all $\boldsymbol{x} \in \mathbb{R}^n$.

The following result shows that log-concavity of $p_{\boldsymbol{x}}$ is sufficient for the convexity of h associated with R_{mmse} .

Proposition 13. The log-concavity of the prior $p_{\boldsymbol{x}}$ implies the log-concavity of $p_{\boldsymbol{s}}$.

Proof. It is well-known that convolution preserves log-concavity. Since both $\mathcal{G}_\sigma$ and $p_{\boldsymbol{x}}$ are log-concave, we have that $p_{\boldsymbol{s}} = (\mathcal{G}_\sigma * p_{\boldsymbol{x}})$ is also log-concave. ■

Remark. From Propositions 12 and 13, we can conclude that

$$p_{\boldsymbol{x}} \text{ is log-concave} \quad \Rightarrow \quad p_{\boldsymbol{s}} \text{ is log-concave} \quad \Leftrightarrow \quad h_{\text{mmse}} \text{ is convex.}$$

Note how log-concavity of $p_{\boldsymbol{x}}$ is not a necessary condition for that of $p_{\boldsymbol{s}}$. This means that there are some priors $p_{\boldsymbol{x}}$ that are not log-concave such that h_{mmse} associated with R_{mmse} is convex.

Deep denoising networks

In practice, denoisers are obtained by training a deep network $\mathsf{R}_{\boldsymbol{\theta}}$ to minimize an empirical loss function

$$\frac{1}{k}\sum_{i=1}^{k} \mathcal{L}(\boldsymbol{x}_i, \mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{s}_i)), \tag{4}$$

where $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k$ are clean images assumed to be sampled from an unknown image prior $p_{\boldsymbol{x}}$ and each $\boldsymbol{s}_i$ is the noisy observation of type (2). Since the loss (4) is an approximation of the true expectation (3), for $\mathcal{L}(\boldsymbol{x}, \widehat{\boldsymbol{x}}) = \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|_2^2$, we can interpret the solution $\mathsf{R}_{\boldsymbol{\theta}^\star}$ of (4) to be an approximation the true MMSE denoiser

$$\mathsf{R}_{\boldsymbol{\theta}^\star}(\boldsymbol{s}) = \mathsf{R}_{\text{mmse}}(\boldsymbol{s}) + \boldsymbol{\varepsilon}(\boldsymbol{s}) = \mathbb{E}[\boldsymbol{x}|\boldsymbol{s}] + \boldsymbol{\varepsilon}(\boldsymbol{s}), \quad \boldsymbol{s} \in \mathbb{R}^n, \tag{5}$$

where we use $\boldsymbol{\varepsilon}(\boldsymbol{s})$ denote the error between the true MMSE and the learned denoisers. While the design and training of state-of-the-art denoisers is beyond the scope of these lectures, interested readers can find plenty of material in the literature [6, 7].

Model-based optimization

It is common to formulate the solutions of inverse problems (1) as an optimization problem of form

$$\widehat{\boldsymbol{x}} \in \operatorname*{arg\,min}_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}) \quad \text{with} \quad f(\boldsymbol{x}) = g(\boldsymbol{x}) + h(\boldsymbol{x}), \tag{6}$$

where g is a data fidelity term that quantifies consistency with the observed measurements $\boldsymbol{y}$ and h is a regularizer that enforces prior knowledge on $\boldsymbol{x}$. The formulation (6) is known as the *variational optimization*, *regularized inversion*, and *model-based reconstruction*.

Remark. The formulation in (6) can be interpreted as the MAP estimator when

$$g(\boldsymbol{x}) = -\log(p_{\boldsymbol{y}|\boldsymbol{x}}(\boldsymbol{x})) \quad \text{and} \quad h(\boldsymbol{x}) = -\log(p_{\boldsymbol{x}}(\boldsymbol{x})), \tag{7}$$

where $p_{\boldsymbol{y}|\boldsymbol{x}}$ is the likelihood relating $\boldsymbol{x}$ to measurements $\boldsymbol{y}$ and $p_{\boldsymbol{x}}$ is the prior distribution.

Example: least squares. One of the most popular functions used in practice is the quadratic function

$$g(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{y} - \mathbf{A}\boldsymbol{x}\|_2^2 = \frac{1}{2}\sum_{i=1}^{m}(\boldsymbol{a}_i^{\mathsf{T}}\boldsymbol{x} - y_i)^2 = \frac{1}{2}\boldsymbol{x}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\mathbf{A}\boldsymbol{x} - \boldsymbol{y}^{\mathsf{T}}\mathbf{A}\boldsymbol{x} + \frac{1}{2}\boldsymbol{y}^{\mathsf{T}}\boldsymbol{y},$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a matrix with m rows $\boldsymbol{a}_i \in \mathbb{R}^n$, $\boldsymbol{x} \in \mathbb{R}^n$ is the optimization variable, and $\boldsymbol{y} \in \mathbb{R}^m$ is the observation vector. Quadratic function above corresponds to the negative log-likelihood for the AWGN. The gradient and the Hessian of the least-squares is given by

$$\nabla g(\boldsymbol{x}) = \mathbf{A}^{\mathsf{T}}(\mathbf{A}\boldsymbol{x} - \boldsymbol{y}) \quad \text{and} \quad \mathsf{H}g(\boldsymbol{x}) = \mathbf{A}^{\mathsf{T}}\mathbf{A} \succeq 0.$$

Example: indicator function. An *indicator function* of a closed set $\mathcal{X} \subseteq \mathbb{R}^n$ corresponds to a regularizer defined as follows

$$\mathbb{1}_{\mathcal{X}}(\boldsymbol{x}) \coloneqq \begin{cases} 0 & \text{if } \boldsymbol{x} \in \mathcal{X} \\ +\infty & \text{if } \boldsymbol{x} \notin \mathcal{X} \end{cases}.$$

Indicator function is often used for imposing hard constraints on the solution, such as nonnegativity. Note that when $\mathcal{X} \neq \varnothing$ is a convex set, the indicator function is a proper, closed, and convex function.

Example: total variation. Another classical regularizer in computational imaging is the *total variation* (**TV**) function. It has two variants. The *isotropic* TV is defined as follows

$$h_{\text{itv}}(\boldsymbol{x}) = \lambda\sum_{i=1}^{n}\|[\boldsymbol{D}\boldsymbol{x}]_i\|_2 = \lambda\sum_{i=1}^{n}\sqrt{\sum_{j=1}^{d}([\boldsymbol{D}_j\boldsymbol{x}]_i)^2},$$

while the *anisotropic* TV is defined as

$$r_{\text{atv}}(\boldsymbol{x}) = \lambda\sum_{i=1}^{n}\|[\boldsymbol{D}\boldsymbol{x}]_i\|_1 = \lambda\sum_{i=1}^{n}\sum_{j=1}^{d}|[\boldsymbol{D}_j\boldsymbol{x}]_i|\,,$$

where $\lambda > 0$ is controls the strength of the regularization and $\boldsymbol{D} = (\boldsymbol{D}_1, \ldots, \boldsymbol{D}_d)$ denotes the d-dimensional discrete gradient. Both variants of TV are nonsmooth convex functions.

Plug-and-play priors

There have been many algorithms developed over the years for solving composite optimization functions of form (6). *Proximal algorithms* have received significant attention due to their ability to solve (6) when g or h is nonsmooth [8]. One important property of proximal algorithms is that they does not explicitly require knowledge of h, relying instead on the prox_h. We know that both MAP and MMSE estimators correspond to proximal operators of some h, motivating us to simply replace prox_h in proximal algorithms by $\mathsf{R}_{\boldsymbol{\theta}}$.

Alternating direction method of multipliers

One of the most widely used and effective proximal algorithms is the *alternating direction method of multipliers* (**ADMM**), which uses an augmented Lagrangian formulation to allow for alternating minimization of each function in turn (see [9] for an overview of ADMM). To develop the algorithm, we consider the following optimization problem over $(\boldsymbol{x}, \boldsymbol{z})$ equivalent to the original problem

$$\mathsf{minimize} \quad g(\boldsymbol{z}) + h(\boldsymbol{x}) \quad \mathsf{subject\ to} \quad \boldsymbol{z} = \boldsymbol{x}. \tag{8}$$

This process of introducing additional convenient variables is known as *variable splitting*. We will denote the optimal value of the problem by

$$f^* = \mathsf{inf}\{g(\boldsymbol{z}) + h(\boldsymbol{x}) : \boldsymbol{z} - \boldsymbol{x} = \boldsymbol{0}\}.$$

We form the *augmented Lagrangian*

$$\begin{aligned} L_\gamma(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{\mu}) &= g(\boldsymbol{z}) + h(\boldsymbol{x}) + \boldsymbol{\mu}^{\mathsf{T}}(\boldsymbol{z} - \boldsymbol{x}) + \frac{1}{2\gamma}\|\boldsymbol{z} - \boldsymbol{x}\|_2^2 \\ &= g(\boldsymbol{z}) + h(\boldsymbol{x}) + \frac{1}{2\gamma}\|\boldsymbol{z} - \boldsymbol{x} + \gamma\boldsymbol{\mu}\|_2^2 - \frac{\gamma}{2}\|\boldsymbol{\mu}\|_2^2, \end{aligned}$$

where $\gamma > 0$ is the quadratic parameter and $\boldsymbol{\mu} \in \mathbb{R}^n$ is the Lagrangian. The augmented Lagrangian can be viewed as a traditional Lagrangian associated with the problem

$$\mathsf{minimize} \quad g(\boldsymbol{z}) + h(\boldsymbol{x}) + \frac{1}{2\gamma}\|\boldsymbol{z} - \boldsymbol{x}\|_2^2 \quad \mathsf{subject\ to} \quad \boldsymbol{z} = \boldsymbol{x}.$$

This problem is equivalent to the original problem, since for any feasible $(\boldsymbol{z}, \boldsymbol{x})$, the term added to the objective is zero. We can re-write the augmented Lagrangian by introducing a *scaled dual variable* $\boldsymbol{s} := \gamma\boldsymbol{\mu}$, which leads to

$$L_\gamma(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{s}) = g(\boldsymbol{z}) + h(\boldsymbol{x}) + \frac{1}{2\gamma}\|\boldsymbol{z} - \boldsymbol{x} + \boldsymbol{s}\|_2^2 - \frac{1}{2\gamma}\|\boldsymbol{s}\|_2^2.$$

We next solve this problem using the *method of multipliers* that has the following form

$$\begin{aligned} (\boldsymbol{z}^t, \boldsymbol{x}^t) &= \underset{\boldsymbol{x},\boldsymbol{z}}{\mathsf{arg\,min}}\ L_\gamma(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{s}^{t-1}) \\ \boldsymbol{s}^t &= \boldsymbol{s}^{t-1} + (\boldsymbol{z}^t - \boldsymbol{x}^t). \end{aligned}$$

The difficulty of running this algorithm in practice lies in the first step, where there is a need to jointly minimize over both $\boldsymbol{z}$ and $\boldsymbol{x}$. ADMM splits this step into two, substantially simplifying the optimization.

$$\begin{aligned} \boldsymbol{z}^t &= \underset{\boldsymbol{z}\in\mathbb{R}^n}{\mathsf{arg\,min}}\ L_\gamma(\boldsymbol{z}, \boldsymbol{x}^{t-1}, \boldsymbol{s}^{t-1}) = \mathsf{prox}_{\gamma g}(\boldsymbol{x}^{t-1} - \boldsymbol{s}^{t-1}) \\ \boldsymbol{x}^t &= \underset{\boldsymbol{x}\in\mathbb{R}^n}{\mathsf{arg\,min}}\ L_\gamma(\boldsymbol{z}^t, \boldsymbol{x}, \boldsymbol{s}^{t-1}) = \mathsf{prox}_{\gamma h}(\boldsymbol{z}^t + \boldsymbol{s}^{t-1}) \\ \boldsymbol{s}^t &= \boldsymbol{s}^{t-1} + (\boldsymbol{z}^t - \boldsymbol{x}^t). \end{aligned}$$

PnP-ADMM [10, 11] enables the substitution of prox_h by a deep denoiser $\mathsf{R}_{\boldsymbol{\theta}}$.

Plug-and-play ADMM (PnP-ADMM)

Input: An initial values $\boldsymbol{x}^0 \in \mathbb{R}^n$ and $\boldsymbol{s}^0 = \boldsymbol{0}$, and a parameter $\gamma > 0$.

Iterate: For $t = 1, 2, 3, \ldots,$ do

$$\begin{aligned}\boldsymbol{z}^t &\leftarrow \mathsf{prox}_{\gamma g}(\boldsymbol{x}^{t-1} - \boldsymbol{s}^{t-1})\\ \boldsymbol{x}^t &\leftarrow \mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{z}^t + \boldsymbol{s}^{t-1})\\ \boldsymbol{s}^t &\leftarrow \boldsymbol{s}^{t-1} + (\boldsymbol{z}^t - \boldsymbol{x}^t).\end{aligned}$$

Remark. PnP-ADMM has an important feature of modularity; it explicitly separate the application of the physical models (the data fidelity update) from that of the learned models (the image denoising). This observation reveals a key strength of PnP methods: they can be easily customized for different measurement operators by changing the data fidelity term, thus enabling the use of the same learned network over a wide range of applications without retraining.

It is possible to establish the convergence of ADMM under some basic assumptions.

Proposition 14. Let $g \in \Gamma^0(\mathbb{R}^n)$ and $h \in \Gamma^0(\mathbb{R}^m)$ and assume that the unaugmented Lagrangian L_0 has a saddle point. Then, ADMM iterates satisfy the following:

- *Residual convergence*: The iterates approach feasibility

$$\lim_{t\to\infty} (\boldsymbol{z}^t - \boldsymbol{x}^t) = \boldsymbol{0}.$$

- *Objective convergence*: The objective function of the iterates approaches the optimum

$$\lim_{t\to\infty} \big(g(\boldsymbol{z}^t) + h(\boldsymbol{x}^t)\big) = f^*.$$

- *Dual variable convergence*: Lagrangian converges to the dual optimal point

$$\lim_{t\to\infty} \boldsymbol{\mu}^t = \boldsymbol{\mu}^*.$$

Remark. The unaugmented Lagrangian L_0 has a saddle point if there exist $(\boldsymbol{z}^*, \boldsymbol{x}^*, \boldsymbol{s}^*)$ for which

$$L_0(\boldsymbol{z}^*, \boldsymbol{x}^*, \boldsymbol{s}) \leq L_0(\boldsymbol{z}^*, \boldsymbol{x}^*, \boldsymbol{s}^*) \leq L_0(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{s}^*), \quad \forall(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{s}).$$

Together, the convexity and the saddle point condition imply that the strong duality will hold for the constrained optimization problem. This will imply that $(\boldsymbol{z}^*, \boldsymbol{x}^*)$ is a solution to the problem such that $\boldsymbol{z} - \boldsymbol{x} = \boldsymbol{0}$, which means that $g(\boldsymbol{x}^*) < \infty$ and $h(\boldsymbol{x}^*) < \infty$.

Proof. See Section 3.2 in [9]. ■

Although, the analysis of ADMM above assumes that both g and h are proper, closed, and convex, it can be extend in numerous ways to be more compatible with PnP. For example, one can get the convergence specifically for the MMSE denoisers without any assumptions on the convexity of g and h.

Proposition 15. Assume that $p_{\boldsymbol{x}}$ is non-degenerate over $\mathbb{R}^n$, g is continuously differentiable, h associated with $\mathsf{R}_{\boldsymbol{\theta}} = \mathsf{R}_{\mathsf{mmse}}$ has a M-Lipschitz continuous gradient over $\mathcal{X} := \mathsf{Im}(\mathsf{R}_{\boldsymbol{\theta}})$. Then, PnP-ADMM iterates for $0 < \gamma \leq 1/(2M)$ satisfy $\|\nabla f(\boldsymbol{x}^t)\|_2 \to 0$ as $t \to \infty$.

Proof. See Theorem 1 in [12]. ■

Half-quadratic splitting

Deep plug-and-play image restoration (**DPIR**) is one of the state-of-the-art implementations of PnP based on the DRUNet denoiser [7]. DPIR iterations are not based on ADMM. They instead correspond to the well-known *half-quadratic splitting* (**HQS**) method that seeks to solve (8) using a quadratic penalty method

$$L_\gamma(\boldsymbol{z}, \boldsymbol{x}) = g(\boldsymbol{z}) + h(\boldsymbol{x}) + \frac{1}{2\gamma}\|\boldsymbol{z} - \boldsymbol{x}\|_2^2.$$

DPIR minimizes L_γ by changing the noise level σ parameter of DRUNet at every iteration in a way that corresponds to decreasing the value of γ. In addition to $\gamma > 0$ (denoted as $\mu = 1/\gamma$) The presentation of the original DPIR explicitly includes the noise level $\sigma > 0$ and a parameter $\lambda > 0$ for the regularizer. However, one can conceptually simplify the algorithm as follows by absorbing the parameters into functions g and h.

Deep plug-and-play image restoration (DPIR)

Input: An initial values $\boldsymbol{x}^0 \in \mathbb{R}^n$ and a decreasing set of positive parameters $\{\gamma_t\}$.

Iterate: For $t = 1, 2, 3, \ldots,$ do

$$\boldsymbol{z}^t \leftarrow \mathsf{prox}_{\gamma_t g}(\boldsymbol{x}^{t-1})$$
$$\boldsymbol{x}^t \leftarrow \mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{z}^t, \gamma_t)$$

Proximal gradient method

Another widely-used proximal algorithm in the context of PnP is *proximal gradient method* (**PGM**), which is also known as *forward backward splitting* and *iterative shrinkage/thresholding algorithm*.

Plug-and-play proximal gradient method (PnP-PGM)

Input: An initial value $\boldsymbol{x}^0 \in \mathbb{R}^n$ and a step-size $\gamma > 0$.

For $t = 1, 2, 3, \ldots,$ do

$$\boldsymbol{x}^t \leftarrow \mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{x}^{t-1} - \gamma \nabla g(\boldsymbol{x}^{t-1})).$$

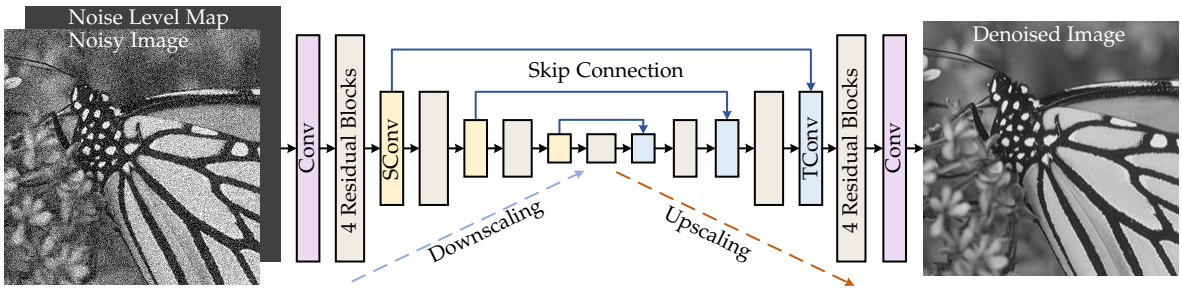


Figure 2: DRUNet is a deep denoiser for DPIR that takes a noisy image $\boldsymbol{s}$ and the noise level σ as inputs [7].

Plug-and-play accelerated PGM (PnP-APGM)

Input: An initial value $\boldsymbol{x}^0 = \boldsymbol{s}^0 \in \mathbb{R}^n$, a step-size $\gamma > 0$, and a set of positive parameters $\{\theta_t\}$.

For $t = 1, 2, 3, \ldots,$ do

$$\boldsymbol{x}^t \leftarrow \mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{s}^{t-1} - \gamma \nabla g(\boldsymbol{s}^{t-1}))$$
$$\boldsymbol{s}^t \leftarrow (1-\theta_t)\boldsymbol{x}^t + \theta_t \boldsymbol{x}^{t-1}$$

Remark. Note the difference in the treatment of data fidelity term g between different PnP algorithm. While PnP-PGM uses the standard (explicit) gradient step $\boldsymbol{z} = \boldsymbol{x} - \gamma \nabla g(\boldsymbol{x})$, PnP-ADMM uses the proximal operator $\mathsf{prox}_{\gamma g}$, which can be written as an implicit gradient step $\boldsymbol{z} = \boldsymbol{x} - \gamma \nabla g(\boldsymbol{z})$ with ∇g evaluated at $\boldsymbol{z}$. For the quadratic loss $g(\boldsymbol{x}) = (1/2)\|\boldsymbol{y} - \mathbf{A}\boldsymbol{x}\|_2^2$, we obtain the following updates

$$\boldsymbol{x} - \gamma g(\boldsymbol{x}) = \mathbf{A}^{\mathsf{T}}(\mathbf{A}\boldsymbol{x} - \boldsymbol{y})$$
$$\mathsf{prox}_{\gamma g}(\boldsymbol{x}) = (\mathbf{I} + \gamma \mathbf{A}^{\mathsf{T}}\mathbf{A})^{-1}(\boldsymbol{x} + \gamma \mathbf{A}^{\mathsf{T}}\boldsymbol{y}).$$

The convergence of both PGM and APGM have been extensively analyzed in the literature.

Theorem 1. Let $f = g + h$ have a finite minimum f^*, g have a L-Lipschitz continuous gradient, and h be sub-differentiable. Run PGM using a fixed step-size $0 < \gamma < 1/L$. Then, the sequence $\{f(\boldsymbol{x}^t)\}$ is non-increasing and $\|\boldsymbol{w}(\boldsymbol{x}^t)\|_2 \to 0$ as $t \to \infty$ with $\boldsymbol{w}(\boldsymbol{x}^t) \in \partial f(\boldsymbol{x}^t)$.

Remark. Note that since h associated with $\mathsf{R}_{\mathsf{mmse}}$ is differentiable on the iterates of PnP-PGM, we have that $\boldsymbol{w}(\boldsymbol{x}^t) = \nabla f(\boldsymbol{x}^t) = \nabla g(\boldsymbol{x}^t) + \nabla h(\boldsymbol{x}^t)$.

Proof. Without loss of generality set $\gamma = 1/(\alpha L)$ with $\alpha > 1$. Consider a single iteration of PGM

$$\boldsymbol{x}^+ \in \mathsf{prox}_{\gamma h}(\boldsymbol{x} - \gamma \nabla g(\boldsymbol{x})) = \operatorname*{arg\,min}_{\boldsymbol{z}} \left\{ \frac{1}{2\gamma}\|\boldsymbol{z} - \boldsymbol{x}\|_2^2 + \nabla g(\boldsymbol{x})^{\mathsf{T}}(\boldsymbol{z} - \boldsymbol{x}) + h(\boldsymbol{z}) \right\}.$$

- From the Lipschitz continuity of ∇g, we have

$$g(\boldsymbol{x}^+) \leq g(\boldsymbol{x}) + \nabla g(\boldsymbol{x})^{\mathsf{T}}(\boldsymbol{x}^+ - \boldsymbol{x}) + \frac{L}{2}\|\boldsymbol{x}^+ - \boldsymbol{x}\|_2^2.$$

- From the first order optimality of prox, we have

$$h(\boldsymbol{x}^+) \leq h(\boldsymbol{x}) - \nabla g(\boldsymbol{x})^{\mathsf{T}}(\boldsymbol{x}^+ - \boldsymbol{x}) - \frac{1}{2\gamma}\|\boldsymbol{x}^+ - \boldsymbol{x}\|_2^2.$$

By combining these two inequalities, we obtain

$$f(\boldsymbol{x}^+) \leq f(\boldsymbol{x}) - (\alpha - 1)\frac{L}{2}\|\boldsymbol{x}^+ - \boldsymbol{x}\|_2^2 \quad \Leftrightarrow \quad \|\boldsymbol{x}^+ - \boldsymbol{x}\|_2^2 \leq \frac{2}{L(\alpha-1)}(f(\boldsymbol{x}) - f(\boldsymbol{x}^+)).$$

Note that this inequality directly implies that $f(\boldsymbol{x}^+) \leq f(\boldsymbol{x})$.

On the other hand, from the first-order optimality conditions for the prox, we know that

$$\mathbf{0} \in \frac{1}{\gamma}(\boldsymbol{x}^+ - \boldsymbol{x}) + \nabla g(\boldsymbol{x}) + \partial h(\boldsymbol{x}^+) \quad \Rightarrow \quad \frac{1}{\gamma}(\boldsymbol{x} - \boldsymbol{x}^+) \in \nabla g(\boldsymbol{x}) + \partial h(\boldsymbol{x}^+).$$

This directly implies that the following vector is a subgradient of $f = g + h$ at $\boldsymbol{x}^+$

$$\boldsymbol{w}(\boldsymbol{x}^+) = \frac{1}{\gamma}(\boldsymbol{x} - \boldsymbol{x}^+) + \nabla g(\boldsymbol{x}^+) - \nabla g(\boldsymbol{x}) \in \partial f(\boldsymbol{x}^+).$$

This implies that

$$\|\boldsymbol{w}(\boldsymbol{x}^+)\|_2 \leq \frac{1}{\gamma}\|\boldsymbol{x}^+ - \boldsymbol{x}\|_2 + \|\nabla g(\boldsymbol{x}^+) - \nabla g(\boldsymbol{x})\|_2 \leq L(\alpha+1)\|\boldsymbol{x}^+ - \boldsymbol{x}\|_2,$$

where we used the triangular inequality and L-Lipschitz continuity of ∇g.

We can thus obtain this bound

$$\|\boldsymbol{w}^+\|_2^2 \leq L^2(\alpha+1)^2\|\boldsymbol{x}^+ - \boldsymbol{x}\|^2 \leq L\frac{(\alpha+1)^2}{(\alpha-1)}(f(\boldsymbol{x}) - f(\boldsymbol{x}^+)),$$

which we can iterate to obtain

$$\sum_{i=0}^{t}\|\boldsymbol{w}(\boldsymbol{x}^i)\|_2^2 \leq L\frac{(\alpha+1)^2}{(\alpha-1)}(f(\boldsymbol{x}^0) - f(\boldsymbol{x}^{t+1})) \leq L\frac{(\alpha+1)^2}{(\alpha-1)}(f(\boldsymbol{x}^0) - f^*),$$

which implies that $\boldsymbol{w}(\boldsymbol{x}^t) \to \boldsymbol{0}$ as $t \to \infty$. ■

Remark. We will skip other analyses of PGM and APGM, since they can be readily found elsewhere. The key point to remember is that APGM enables faster $O(1/t^2)$ convergence in terms of f when the function is convex (see for example Nesterov [13]).

Regularization by denoising

Regularization by denoising (**RED**) is a variant of PnP that seeks to use the gradient of the regularizer associated with a given denoiser rather than its proximal operator. It was originally introduced [14] and its theory has been significantly clarified in subsequent papers [15, 16].

Gradient method regularization by denoising (GM-RED)

Input: An initial value $\boldsymbol{x}^0 \in \mathbb{R}^n$, a step-size $\gamma > 0$, and regularization parameter $\tau > 0$.

For $t = 1, 2, 3, \ldots,$ do

$$\boldsymbol{x}^t \leftarrow \boldsymbol{x}^{t-1} - \gamma(\nabla g(\boldsymbol{x}^{t-1}) + \tau(\boldsymbol{x}^{t-1} - \mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{x}^{t-1}))).$$

PGM-RED

Input: An initial value $\boldsymbol{x}^0 \in \mathbb{R}^n$, a step-size $\gamma > 0$, and regularization parameter $\tau > 0$.

For $t = 1, 2, 3, \ldots,$ do

$$\boldsymbol{x}^t \leftarrow \mathsf{prox}_{\gamma g}(\boldsymbol{x}^{t-1} - \gamma\tau(\boldsymbol{x}^{t-1} - \mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{x}^{t-1}))).$$

Propositions 8 and 10 provide theoretical background necessary to understand the convergence of the RED algorithms for MAP and MMSE denoisers.

Remark. RED seeks to minimize a function $f = g + h_\mu$ where h_μ depends on the type of the denoiser. For simplicity, we will let $\tau = 1/\mu$, where $\mu := \sigma^2$.

- **MAP denoiser:** Consider a MAP denoiser for a log-concave prior $p_{\boldsymbol{x}}$

$$\mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \mathsf{prox}_{\mu h_{\mathsf{map}}}(\boldsymbol{x}) \quad \text{with} \quad h_{\mathsf{map}}(\boldsymbol{x}) := -\log p_{\boldsymbol{x}}(\boldsymbol{x}) \quad \Rightarrow \quad h_\mu \text{ is the Moreau envelope of } h_{\mathsf{map}}$$

- **MMSE denoiser:** Consider a MMSE denoiser

$$\mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \mathsf{R}_{\mathsf{mmse}}(\boldsymbol{x}) \quad \Rightarrow \quad h_\mu(\boldsymbol{x}) = \nabla \log p_{\boldsymbol{s}}(\boldsymbol{x}) = \nabla \log \left(\mathcal{G}_\sigma * p_{\boldsymbol{x}}\right)(\boldsymbol{x}).$$

Gradient-step PnP

The traditional PnP strategy is to use a deep network $\mathsf{R}_{\boldsymbol{\theta}} : \mathbb{R}^n \to \mathbb{R}^n$ to parametrize a MAP or MMSE denoisers, thus giving us an implicit access to a regularizer h that depends on the prior $p_{\boldsymbol{x}}$. An alternative approach is to use a deep network $h_{\boldsymbol{\theta}} : \mathbb{R}^n \to \mathbb{R}$ to parametrize the regularizer directly. An innovative approach is to define the following denoiser for over $\boldsymbol{x} \in \mathbb{R}^n$ [18, 17]

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) := \frac{1}{2}\|\boldsymbol{x} - \mathsf{D}_{\boldsymbol{\theta}}(\boldsymbol{x})\|_2^2 \quad \Rightarrow \quad \mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{x}) := \boldsymbol{x} - \gamma \nabla h_{\boldsymbol{\theta}}(\boldsymbol{x}),$$

where $\mathsf{D}_{\boldsymbol{\theta}} : \mathbb{R}^n \to \mathbb{R}^n$ is a deep network (such as DRUNet or DnCNN). The operator $\mathsf{R}_{\boldsymbol{\theta}}$ is the *gradient-step* (**GS**) denoiser that can be trained and deployed as any traditional PnP denoiser. Two key benefit of the GS approach is that (a) the denoiser exactly represents a conservative vector field and (b) there is a direct access to the regularizer $h_{\boldsymbol{\theta}}$.

Online PnP methods

The traditional PnP methods are batch algorithms in the sense that they compute the gradient ∇g or the proximal operator $\mathsf{prox}_{\gamma g}$ of the data fidelity term g by using the whole measurement vector $\boldsymbol{y} \in \mathbb{R}^m$. The per-iteration computational and memory complexity of batch PnP algorithms depends on the total number of measurements. For example, in tomography with b projections, the complexity of evaluating ∇g scales linearly with b, making it computationally expensive for a large b. This has motivated interest in *online*, *stochastic*, and/or *incremental* PnP algorithms that approximate the batch ∇g with an approximation $\hat{\nabla} g$ based on a single element or a small subset of the measurements [19, 20, 21].

Consider the decomposition of $\mathbb{R}^m$ into $b \geq 1$ blocks

$$\mathbb{R}^m = \mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \times \cdots \times \mathbb{R}^{m_b} \quad \text{with} \quad m = m_1 + m_2 + \cdots + m_b.$$

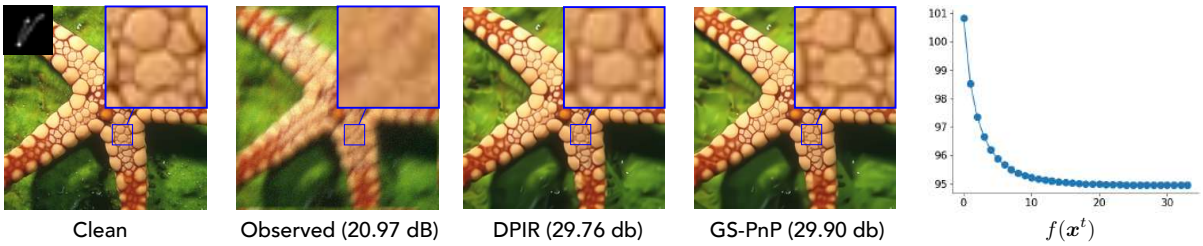


Figure 3: The strategy of explicitly parametrizing the regularizer $h_{\boldsymbol{\theta}}$ in GS-PnP provides a competitive performance relative to DPIR, which is a state-of-the-art PnP method [17].

In this setting, the data fidelity term is given by

$$g(\boldsymbol{x}) = \frac{1}{b}\sum_{i=1}^{b} g_i(\boldsymbol{x})$$

where each g_i is evaluated only on the subset $\boldsymbol{y}_i \in \mathbb{R}^{m_i}$ of the full measurement vector $\boldsymbol{y} \in \mathbb{R}^m$. For example, each individual term can be set to the quadratic function $g_i(\boldsymbol{x}) = (1/2)\|\boldsymbol{y}_i - \mathbf{A}_i\boldsymbol{x}\|_2^2$, where $\mathbf{A}_i$ is the operator corresponding to the measurement block $\boldsymbol{y}_i$. Online PnP algorithms improve the scalability to large-scale measurements by using only a single component gradient $\nabla g_i(\boldsymbol{x})$ or a single component proximal operator $\mathsf{prox}_{\gamma g_i}(\boldsymbol{x})$ with $i \in \{1, \ldots, b\}$,making their per-iteration complexity independent of b.

Online PnP algorithms can be implemented using different block selection rules. The strategy commonly adopted for the theoretical analysis focuses on selecting indices i_t as independent identically distributed random variables distributed uniformly over $\{1, \ldots, b\}$. An alternative would be to proceed in epochs of b consecutive iterations, where the set $\{1, \ldots, b\}$ is reshuffled at the start of each epoch and the index i_t is selected from this ordered set.

Online PnP-PGM

Input: An initial value $\boldsymbol{x}^0 \in \mathbb{R}^n$ and a step-size $\gamma > 0$.

For $t = 1, 2, 3, \ldots,$ do

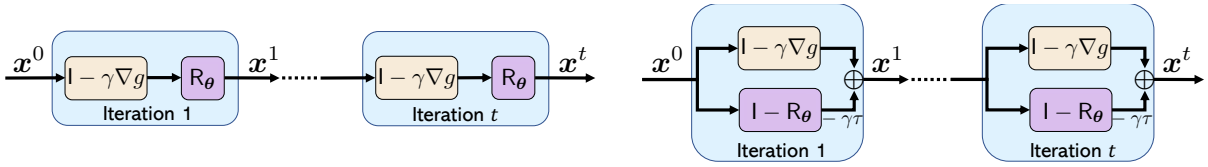
$$\text{Choose an index } i_t \in \{1, \ldots, b\}$$
$$\boldsymbol{x}^t \leftarrow \mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{x}^{t-1} - \gamma \nabla g_{i_t}(\boldsymbol{x}^{t-1}))$$

Incremental plug-and-play ADMM (IPA)

Input: An initial values $\boldsymbol{x}^0 \in \mathbb{R}^n$ and $\boldsymbol{s}^0 = \mathbf{0}$, and a parameter $\gamma > 0$.

Iterate: For $t = 1, 2, 3, \ldots,$ do

$$\text{Choose an index } i_t \in \{1, \ldots, b\}$$
$$\boldsymbol{z}^t \leftarrow \mathsf{prox}_{\gamma g_{i_t}}(\boldsymbol{x}^{t-1} - \boldsymbol{s}^{t-1})$$
$$\boldsymbol{x}^t \leftarrow \mathsf{R}_{\boldsymbol{\theta}}(\boldsymbol{z}^t + \boldsymbol{s}^{t-1})$$
$$\boldsymbol{s}^t \leftarrow \boldsymbol{s}^{t-1} + (\boldsymbol{z}^t - \boldsymbol{x}^t).$$



(a) Deep unfolding of PnP-PGM

(b) Deep unfolding of GM-RED

Figure 4: PnP is related to two other popular computational-imaging paradigms, *deep unfolding* (**DU**) and *deep equilibrium models* (**DEQ**). PnP-PGM or GM-RED can be turned into a DU architecture by truncating the algorithm to $t \geq 1$ iterations and training the weights $\boldsymbol{\theta}$ of the network $\mathsf{R}_{\boldsymbol{\theta}}$ end-to-end. Similarly, a DEQ architecture can be obtained by running the PnP algorithm until convergence and using the implicit differentiation at the fixed point to train $\boldsymbol{\theta}$. The operator $\mathsf{R}_{\boldsymbol{\theta}}$ in DU/DEQ is not necessarily an AWGN denoiser, instead it is an *artifact-removal* (**AR**) operator trained to remove artifacts specific to the PnP iterations.

Deep unfolding and deep equilibrium learning

Deep unfolding (**DU**) (also known as *deep unrolling* or *algorithm unrolling*) is a DL paradigm that has gained popularity in computational imaging due to its ability to provide a systematic connection between iterative algorithms and deep neural network architectures. PnP algorithms can be easily turned into DU architectures by parameterizing the operator $\mathsf{R}_{\boldsymbol{\theta}}$ as a CNN with weights $\boldsymbol{\theta}$, truncating the PnP algorithm to a fixed number of iterations $t \geq 1$, and training the corresponding architecture end-to-end in a supervised fashion. For example, Figure 4 illustrates the representation of t iterations of PnP-ISTA and RED-SD as DU architectures.

Consider a set of paired data $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, where $\boldsymbol{x}_i$ is the desired "ground truth" image and $\boldsymbol{y}_i = \mathbf{A}\boldsymbol{x}_i + \boldsymbol{e}_i$ is its noisy observation. Consider also the iterate $\boldsymbol{x}_i^t(\boldsymbol{\theta})$ of a PnP algorithm truncated to $t \geq 1$ iterations, where we made explicit the dependence of the PnP output on the weights $\boldsymbol{\theta}$ of the deep network parameterizing $\mathsf{R}_{\boldsymbol{\theta}}$. DU interprets the steps required for mapping the input vector $\boldsymbol{y}_i$ and the initialization $\boldsymbol{x}_i^0$ to the output $\boldsymbol{x}_i^t(\boldsymbol{\theta})$ as layers of a deep neural network architecture. The DU training is performed by solving

$$\widehat{\boldsymbol{\theta}} \in \operatorname*{arg\,min}_{\boldsymbol{\theta}} \sum_i \mathcal{L}(\boldsymbol{x}_i, \boldsymbol{x}_i^t(\boldsymbol{\theta}))\,, \tag{9}$$

where $\mathcal{L}$ is a loss function that quantifies the discrepancy between the true and predicted solutions. Once trained using (9), the truncated PnP algorithm can be directly used for imaging [22].

Deep equilibrium models (DEQ) is an extension of DU to an arbitrary number of iterations [23]. DEQ can be implemented by replacing $\boldsymbol{x}_i^t(\boldsymbol{\theta})$ in (9) by a fixed-point $\overline{\boldsymbol{x}}_i(\boldsymbol{\theta})$ of a given PnP algorithm and using implicit differentiation for updating the weights $\boldsymbol{\theta}$. The benefit of DEQ over DU is that it doesn't require the storage of the intermediate variables for solving (9), thus reducing the memory complexity of training. However, DEQ requires the computation of the fixed-point $\overline{\boldsymbol{x}}_i(\boldsymbol{\theta})$, which can increase the computational complexity.

There are some important differences between traditional PnP and DU/DEQ. Traditional PnP relies on an AWGN denoiser as an image prior. On the other hand, the operator $\mathsf{R}_{\boldsymbol{\theta}}$ in DU/DEQ is not an AWGN denoiser; instead, it is an *artifact-removal* (**AR**) operator trained to remove artifacts specific to the PnP iterations. As seen in Fig. 5, which shows the relative performance of PnP using an AWGN denoiser and using a pre-trained AR operator, this problem-specific training can yield significantly improved results. However, this performance comes at a cost; while the prior in traditional PnP is *fully decoupled* from the measurement operator, that of DU/DEQ is trained by accounting for the measurement operator $\mathbf{A}$. Hence the DU/DEQ approach has reduced generality and higher computational/memory complexity of training, since the AR prior is trained for the specific task of reconstruction from random projections rather than for AWGN denoising.

Monotone operator theory

Monotone operator theory [24] provides a set of mathematical tools for analyzing convergence of PnP, DU with weight sharing, and DEQ in a unified fashion. The approach is to view the algorithm as a fixed-point

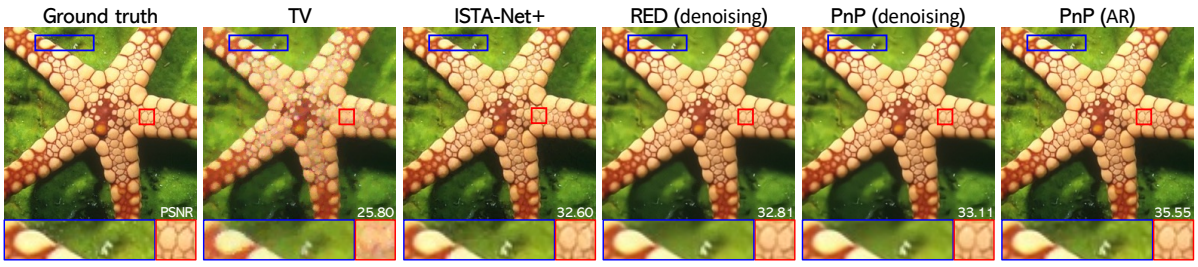


Figure 5: Visual evaluation of color image recovery in compressive sensing from random projections with 20% subsampling. The results of TV and a well-known DU architecture ISTA-Net+ are provided for reference. The methods PnP (denoising) and RED (denoising) use a pre-trained AWGN denoiser as an image prior. The method PnP (AR) uses a problem-dependent AR operator pre-trained using DU. Note that the choice of denoiser affects the reconstruction significantly (PSNR shown in white).

iteration of a contractive or nonexpansive operator.

Example. PnP-ADMM is equivalent to running the following fixed-point iteration

$$\boldsymbol{v}^t \leftarrow \mathsf{T}_{\boldsymbol{\theta}}(\boldsymbol{v}^{t-1}) \quad \text{with} \quad \mathsf{T}_{\boldsymbol{\theta}} \coloneqq \frac{1}{2}\mathsf{I} + \frac{1}{2}(2\mathsf{prox}_{\gamma g} - 1)(2\mathsf{R}_{\boldsymbol{\theta}} - 1),$$

where I is the identity and $\mathsf{prox}_{\gamma g}$ is the proximal operator of g. Similarly, PnP-PGM is equivalent to running the following fixed-point iteration

$$\boldsymbol{v}^t \leftarrow \mathsf{F}_{\boldsymbol{\theta}}(\boldsymbol{v}^{t-1}) \quad \text{with} \quad \mathsf{F}_{\boldsymbol{\theta}} \coloneqq \mathsf{R}_{\boldsymbol{\theta}}(\mathsf{I} - \gamma\nabla g),$$

where ∇g is the gradient of g.

Definition 4. An operator T is *Lipschitz continuous* with constant $\lambda > 0$ if

$$\|\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y}\|_2 \leq \lambda\|\boldsymbol{x} - \boldsymbol{y}\|_2, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n.$$

When $\lambda = 1$, we say that T is *nonexpansive*. When $\lambda < 1$, we say that T is a *contraction*.

Definition 5. T is *monotone* if

$$(\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y})^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y}) \geq 0, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n.$$

We say that it is *strongly monotone* or *coercive* with parameter $\mu > 0$ if

$$(\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y})^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y}) \geq \mu\|\boldsymbol{x} - \boldsymbol{y}\|^2, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n.$$

Definition 6. T is cocoercive with constant $\beta > 0$ if

$$(\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y})^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y}) \geq \beta\|\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y}\|^2, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n.$$

When $\beta = 1$, we say that T is firmly nonexpansive.

Definition 7. For a constant $\alpha \in (0, 1)$, we say that T is α-averaged, if there exists a nonexpansive operator N such that $\mathsf{T} = (1 - \alpha)\mathsf{I} + \alpha\mathsf{N}$.

Using the definitions above, one can show the following

Proposition 16. Let $g : \mathbb{R}^n \to \mathbb{R}$ be a convex function. Then, we have that

$$\nabla g \text{ is Lipschitz continuous with constant } L \quad \Leftrightarrow \quad \nabla g \text{ is cocoercive with constant } (1/L).$$

Proof. This proof will be left to the students as part of the exercises. ■

Example. Let $g : \mathbb{R}^n \to \mathbb{R}$ be a convex function with a L-Lipschitz continuous gradient and $\mathsf{R}_{\boldsymbol{\theta}}$ be a firmly-nonexpansive operator. What can we say about the convergence of PnP-PGM and PnP-ADMM?

Appendix: Monotone operators

The following results are derived from the definition above.

Proposition 17. Consider $\mathsf{R} = \mathsf{I} - \mathsf{T}$ where $\mathsf{T} : \mathbb{R}^n \to \mathbb{R}^n$.

$$\mathsf{T} \text{ is nonexpansive } \Leftrightarrow \mathsf{R} \text{ is } (1/2)\text{-cocoercive.}$$

Proof. First suppose that R is $1/2$ cocoercive. Let $\boldsymbol{h} \coloneqq \boldsymbol{x} - \boldsymbol{y}$ for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. We then have

$$\frac{1}{2}\|\mathsf{R}\boldsymbol{x} - \mathsf{R}\boldsymbol{y}\|^2 \leq (\mathsf{R}\boldsymbol{x} - \mathsf{R}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h} = \|\boldsymbol{h}\|^2 - (\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h}.$$

We also have that

$$\frac{1}{2}\|\mathsf{R}\boldsymbol{x} - \mathsf{R}\boldsymbol{y}\|^2 = \frac{1}{2}\|\boldsymbol{h}\|^2 - (\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h} + \frac{1}{2}\|\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y}\|^2.$$

By combining these two and simplifying the expression

$$\|\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y}\| \leq \|\boldsymbol{h}\|.$$

The converse can be proved by following this logic in reverse. ■

Proposition 18. Consider $\mathsf{R} = \mathsf{I} - \mathsf{T}$ where $\mathsf{T} : \mathbb{R}^n \to \mathbb{R}^n$.

$$\mathsf{T} \text{ is Lipschitz continuous with constant } \lambda < 1 \quad \Rightarrow \quad \mathsf{R} \text{ is } (1-\lambda)\text{-strongly monotone.}$$

Proof. By using the Cauchy-Schwarz inequality, we have for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$

$$(\mathsf{R}\boldsymbol{x} - \mathsf{R}\boldsymbol{y})^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|^2 - (\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y})^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y})$$
$$\geq \|\boldsymbol{x} - \boldsymbol{y}\|^2 - \|\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y}\|\|\boldsymbol{x} - \boldsymbol{y}\| \geq \|\boldsymbol{x} - \boldsymbol{y}\|^2 - \lambda\|\boldsymbol{x} - \boldsymbol{y}\|^2 \geq (1-\lambda)\|\boldsymbol{x} - \boldsymbol{y}\|^2.$$

■

The following characterization is often convenient.

Proposition 19. For a nonexpansive operator T, a constant $\alpha \in (0, 1)$, and the operator $\mathsf{R} \coloneqq \mathsf{I} - \mathsf{T}$, the following are equivalent

(a) T is α-averaged

(b) $(1 - 1/\alpha)\mathsf{I} + (1/\alpha)\mathsf{T}$ is nonexpansive

(c) $\|\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y}\|^2 \leq \|\boldsymbol{x} - \boldsymbol{y}\|^2 - \left(\frac{1-\alpha}{\alpha}\right)\|\mathsf{R}\boldsymbol{x} - \mathsf{R}\boldsymbol{y}\|^2$, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$.

Proof. See Proposition 4.35 in [25]. ■

Proposition 20. Consider $\mathsf{T} : \mathbb{R}^n \to \mathbb{R}^n$ and $\beta > 0$. Then, the following are equivalent

(a) T is β-cocoercive

(b) $\beta\mathsf{T}$ is firmly nonexpansive

(c) $\mathsf{I} - \beta\mathsf{T}$ is firmly nonexpansive.

(d) $\beta\mathsf{T}$ is $(1/2)$-averaged.

(e) $\mathsf{I} - 2\beta\mathsf{T}$ is nonexpansive.

Proof. For any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, let $\boldsymbol{h} \coloneqq \boldsymbol{x} - \boldsymbol{y}$. The equivalence between (a) and (b) is readily observed by defining $\mathsf{P} \coloneqq \beta\mathsf{T}$ and noting that

$$\begin{aligned}(\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h} &= \beta(\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h} \quad \text{and} \\ \|\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y}\|^2 &= \beta^2\|\mathsf{T}\boldsymbol{x} - \mathsf{T}\boldsymbol{y}\|.\end{aligned}$$

Define $\mathsf{R} \coloneqq \mathsf{I} - \mathsf{P}$ and suppose (b) is true, then

$$\begin{aligned}&(\mathsf{R}\boldsymbol{x} - \mathsf{R}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h} \\ &= \|\boldsymbol{h}\|^2 - (\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h} \\ &= \|\mathsf{R}\boldsymbol{x} - \mathsf{R}\boldsymbol{y}\|^2 + (\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h} - \|\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y}\|^2 \\ &\geq \|\mathsf{R}\boldsymbol{x} - \mathsf{R}\boldsymbol{y}\|^2.\end{aligned}$$

By repeating the same argument for $\mathsf{P} = \mathsf{I} - \mathsf{R}$, we establish the full equivalence between (b) and (c).

The equivalence of (b) and (d) can be seen by noting that

$$\begin{aligned}&2\|\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y}\|^2 \leq 2(\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h} \\ \Leftrightarrow \quad &\|\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y}\|^2 \leq 2(\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h} - \|\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y}\|^2 \\ &= \|\boldsymbol{h}\|^2 - (\|\boldsymbol{h}\|^2 - 2(\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y})^{\mathsf{T}}\boldsymbol{h} + \|\mathsf{P}\boldsymbol{x} - \mathsf{P}\boldsymbol{y}\|^2) \\ &= \|\boldsymbol{h}\|^2 - \|\mathsf{R}\boldsymbol{x} - \mathsf{R}\boldsymbol{y}\|^2.\end{aligned}$$

To show the equivalence with (e), first suppose that $\mathsf{N} \coloneqq \mathsf{I} - 2\mathsf{P}$ is nonexpansive, then $\mathsf{P} = \frac{1}{2}(\mathsf{I} + (-\mathsf{N}))$ is $1/2$-averaged, which means that it is firmly nonexpansive. On the other hand, if P is firmly nonexpansive, then it is $1/2$-averaged, which means that from Proposition 19(b) we have that $(1-2)\mathsf{I} + 2\mathsf{P} = 2\mathsf{P} - \mathsf{I} = -\mathsf{N}$ is nonexpansive. This directly means that N is nonexpansive. ■

Proposition 21. Let f be a proper, closed, and convex function. Then for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, $\boldsymbol{g} \in \partial f(\boldsymbol{x})$, and $\boldsymbol{h} \in \partial f(\boldsymbol{y})$, ∂f is a monotone operator

$$(\boldsymbol{g} - \boldsymbol{h})^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y}) \geq 0.$$

Additionally if f is strongly convex with constant $\mu > 0$, then ∂f is strongly monotone with the same constant.

$$(\boldsymbol{g} - \boldsymbol{h})^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y}) \geq \mu\|\boldsymbol{x} - \boldsymbol{y}\|^2.$$

Proof. Consider a strongly convex function f with a constant $\mu \geq 0$. Then, we have that

$$\begin{aligned}&\begin{cases} f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \boldsymbol{g}^{\mathsf{T}}(\boldsymbol{y} - \boldsymbol{x}) + \frac{\mu}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2 \\ f(\boldsymbol{x}) \geq f(\boldsymbol{y}) + \boldsymbol{h}^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y}) + \frac{\mu}{2}\|\boldsymbol{x} - \boldsymbol{y}\|^2 \end{cases} \\ \Rightarrow \quad &(\boldsymbol{g} - \boldsymbol{h})^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y}) \geq \mu\|\boldsymbol{x} - \boldsymbol{y}\|^2.\end{aligned}$$

The proof for a weakly convex f is obtained by considering $\mu = 0$ in the inequalities above. ■

Appendix: Subgradient and subdifferential

This chapter reviews the concept of *subgradient* and *subdifferential* that generalize that of the gradient.

Definition and technicalities

We start by considering extended real-valued functions and defining some useful properties.

Definition 8. We define the *domain* of function f as the set

$$\mathsf{dom}(f) = \{\boldsymbol{x} \in \mathbb{R}^n : f(\boldsymbol{x}) < \infty\}.$$

We always assume that $\mathsf{dom}(f) \neq \varnothing$.

Definition 9. We say that a function $f : \mathbb{R}^n \to [-\infty, \infty]$ is *proper* if

- $\exists \boldsymbol{x} \in \mathbb{R}^n$ such that $f(\boldsymbol{x}) < \infty$ (*non-empty domain*)
- $\forall \boldsymbol{x} \in \mathbb{R}^n$ we have $f(\boldsymbol{x}) > -\infty$. (*never attains* $-\infty$)

Remark. When we use the term *proper*, we imply that the function takes values in the extended real-line $(-\infty, \infty]$ with at least one $\boldsymbol{x} \in \mathbb{R}^n$ such that $f(\boldsymbol{x}) < \infty$.

Definition 10. We say that a function f is *closed*, if for each $\theta \in \mathbb{R}$ the *sublevel set*

$$L_f(\theta) \coloneqq \{\boldsymbol{x} \in \mathbb{R}^n : f(\boldsymbol{x}) \leq \theta\}$$

is closed.

Remark. We will denote the class of all convex, proper, and closed functions with $\Gamma^0(\mathbb{R}^n)$.

Definition 11. Let f be a proper function. The vector $\boldsymbol{g} \in \mathbb{R}^n$ is a *subgradient* of f at $\boldsymbol{x} \in \mathbb{R}^n$ if

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \boldsymbol{g}^{\mathsf{T}}(\boldsymbol{y} - \boldsymbol{x}), \quad \forall \boldsymbol{y} \in \mathsf{dom}(f).$$

We also define the *subdifferential* of f at $\boldsymbol{x} \in \mathbb{R}^n$ as a set

$$\partial f(\boldsymbol{x}) \coloneqq \{\boldsymbol{g} \in \mathbb{R}^n : f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \boldsymbol{g}^{\mathsf{T}}(\boldsymbol{y} - \boldsymbol{x}), \forall \boldsymbol{y} \in \mathsf{dom}(f)\}.$$

We will use the notation $\boldsymbol{g}(\boldsymbol{x})$ for highlighting the dependence of the subgradient on $\boldsymbol{x}$.

Remark. The inequality in Definition 11 is known as the *subgradient inequality*. Thus, if f has a subgradient at $\boldsymbol{x}$, then it has a linear lower bound at that location. Note that the inequality is trivial for $\boldsymbol{y} \notin \mathsf{dom}(f)$. When $\boldsymbol{x} \notin \mathsf{dom}(f)$, we define $\partial f(\boldsymbol{x}) = \varnothing$, since the subgradient inequality does not hold for any $\boldsymbol{y} \in \mathsf{dom}(f)$. We will use the notation $\boldsymbol{g}(\boldsymbol{x}) \in \partial f(\boldsymbol{x})$ when we want to make it explicit that the subgradient is evaluated at $\boldsymbol{x}$.

The next results establishes that the set $\partial f(\boldsymbol{x})$ is closed and convex.

Theorem 2. Let f be a proper function. Then, the set $\partial f(\boldsymbol{x})$ is closed and convex for any $\boldsymbol{x} \in \mathbb{R}^n$.

Proof. For any $\boldsymbol{x} \in \mathbb{R}^n$, the subdifferential can be represented as

$$\partial f(\boldsymbol{x}) = \bigcap_{\boldsymbol{y} \in \mathbb{R}^n} H_{\boldsymbol{y}} \quad \text{with} \quad H_{\boldsymbol{y}} = \{\boldsymbol{g} \in \mathbb{R}^n : f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \boldsymbol{g}^{\mathsf{T}}(\boldsymbol{y} - \boldsymbol{x})\}.$$

Since each $H_{\boldsymbol{y}}$ is a half-space, which means that it is closed and convex, the set $\partial f(\boldsymbol{x})$ is closed and convex since it is an intersection of closed and convex sets. ■

However, $\partial f(\boldsymbol{x})$ may be empty at $\boldsymbol{x}$, in which case we say that f is *not* subdifferentiable there.

Definition 12. A proper function f is *subdifferentiable* at $\boldsymbol{x} \in \mathbb{R}^n$ if $\partial f(\boldsymbol{x}) \neq \varnothing$. We will denote the collection of points of subdifferentiability by

$$\mathsf{dom}(\partial f) := \{\boldsymbol{x} \in \mathbb{R}^n : \partial f(\boldsymbol{x}) \neq \varnothing\}.$$

Convex functions are not necessarily subdifferentiable in their domain; however, they are subdifferentiable at any point in the *interior* of their domain.

Theorem 3. If f be proper and convex, then $\partial f(\boldsymbol{x})$ is nonempty and bounded for any $\boldsymbol{x} \in \mathsf{int}(\mathsf{dom}(f))$.

Proof. This proof is highly technical, but can be found in standard textbooks on convex optimization. ■

Remark. Consider an *open ball* centered at $\boldsymbol{x} \in \mathbb{R}^n$ of radius $\epsilon > 0$

$$\mathcal{B}(\boldsymbol{x}, \varepsilon) := \{\boldsymbol{y} \in \mathbb{R}^n : \|\boldsymbol{x} - \boldsymbol{y}\|_2 < \varepsilon\}$$

A point $\boldsymbol{x} \in \mathbb{R}^n$ is an *interior point* of $\mathsf{dom}(f)$ if

$$\exists \varepsilon > 0 \quad \text{such that} \quad \mathcal{B}(\boldsymbol{x}, \varepsilon) \subset \mathsf{dom}(f),$$

which basically means that there exists an open ball centered at $\boldsymbol{x}$ fully contained in $\mathsf{dom}(f)$.

The direct consequence of Theorem 3 is that real-valued convex functions are subdifferentiable everywhere.

Corollary 2.

$$f : \mathbb{R}^n \to \mathbb{R} \text{ is a convex function} \quad \Rightarrow \quad f \text{ is subdifferentiable over } \mathbb{R}^n.$$

When f is differentiable, the subdifferential and gradient are equivalent.

Theorem 4. Let f be a proper convex function and let $\boldsymbol{x} \in \mathsf{int}(\mathsf{dom}(f))$. Then

$$f \text{ is differentiable at } \boldsymbol{x} \quad \Leftrightarrow \quad \partial f(\boldsymbol{x}) = \{\nabla f(\boldsymbol{x})\}.$$

Proof. We prove the result only for real-valued functions and in the $\Rightarrow$ direction. Let us consider an arbitrary $\boldsymbol{g} \in \partial f(\boldsymbol{x})$. Then, for all $\boldsymbol{u} \in \mathbb{R}^n$ and $\epsilon > 0$, we have from the definition of the subgradient

$$f(\boldsymbol{x} + \epsilon \boldsymbol{u}) \geq f(\boldsymbol{x}) + \epsilon \boldsymbol{g}^{\mathsf{T}} \boldsymbol{u} \quad \Rightarrow \quad \lim_{\epsilon \to 0} \frac{f(\boldsymbol{x} + \epsilon \boldsymbol{u}) - f(\boldsymbol{x})}{\epsilon} := \nabla f(\boldsymbol{x})^{\mathsf{T}} \boldsymbol{u} \geq \boldsymbol{g}^{\mathsf{T}} \boldsymbol{u} \quad \Rightarrow \quad (\nabla f(\boldsymbol{x}) - \boldsymbol{g})^{\mathsf{T}} \boldsymbol{u} \geq 0,$$

where we used the definition of the directional derivative. By choosing $\boldsymbol{u} = \boldsymbol{g} - \nabla f(\boldsymbol{x})$, we obtain

$$-\|\nabla f(\boldsymbol{x}) - \boldsymbol{g}\|_2^2 \geq 0 \quad \Rightarrow \quad \|\nabla f(\boldsymbol{x}) - \boldsymbol{g}\|_2^2 \leq 0 \quad \Rightarrow \quad \boldsymbol{g} = \nabla f(\boldsymbol{x}).$$

■

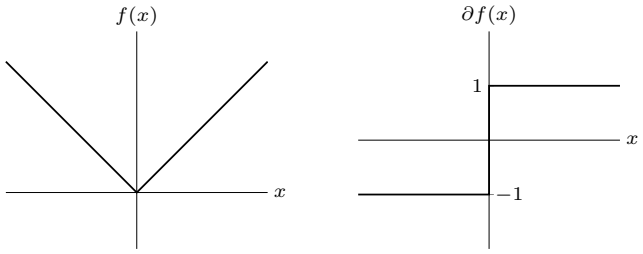


Figure 6: The only point at which $f(x) = |x|$ is not differentiable is $x = 0$, where we have $\partial f(x) = [-1, 1]$.

Example 1. Consider the function $f(x) = |x|$ with $x \in \mathbb{R}$. The only point at which f is not differentiable is $x = 0$. At this point, the subgradients of f are characterized by the inequality

$$f(x) - f(0) \geq g \cdot x \quad \Rightarrow \quad |x| \geq gx \quad \Rightarrow \quad g \in [-1, 1].$$

For $x \neq 0$, the subderivative coincides with the derivative, which means that

$$\partial f(x) = \begin{cases} 1 & \text{if } x > 0 \\ [-1, 1] & \text{if } x = 0 \\ -1 & \text{if } x < 0 \end{cases}.$$

We can also generalize the notion of the *directional derivative* of f.

Theorem 5. Let f be a proper convex function. Then for any $\boldsymbol{x} \in \mathsf{int}(\mathsf{dom}(f))$ and $\boldsymbol{d} \in \mathbb{R}^n$

$$\nabla_{\boldsymbol{d}} f(\boldsymbol{x}) = \lim_{\gamma \to 0^+} \frac{f(\boldsymbol{x} + \gamma \boldsymbol{d}) - f(\boldsymbol{x})}{\gamma} = \mathsf{max}\{\boldsymbol{g}^{\mathsf{T}} \boldsymbol{d} : \boldsymbol{g} \in \partial f(\boldsymbol{x})\}.$$

Proof. We will show the proof for "$\geq$" and omit the proof for "$\leq$". Let $\boldsymbol{x} \in \mathsf{int}(\mathsf{dom}(f))$ and $\boldsymbol{d} \in \mathbb{R}^n$. From the subgradient inequality we have that for any $\boldsymbol{g} \in \partial f(\boldsymbol{x})$

$$\nabla_{\boldsymbol{d}} f(\boldsymbol{x}) = \lim_{\gamma \to 0^+} \frac{f(\boldsymbol{x} + \gamma \boldsymbol{d}) - f(\boldsymbol{x})}{\gamma} \geq \lim_{\gamma \to 0^+} \boldsymbol{g}^{\mathsf{T}} \boldsymbol{d} = \boldsymbol{g}^{\mathsf{T}} \boldsymbol{d} \quad \Rightarrow \quad \nabla_{\boldsymbol{d}} f(\boldsymbol{x}) \geq \mathsf{max}\{\boldsymbol{g}^{\mathsf{T}} \boldsymbol{d} : \boldsymbol{g} \in \partial f(\boldsymbol{x})\}.$$

■

References

[1] 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," *IEEE Signal Process. Mag.*, vol. 40, no. 1, pp. 85–97, Jan. 2023.

[2] J. J. Moreau, "Proximité et dualité dans un espace hilbertien," *Bull. Soc. Math. France*, vol. 93, pp. 273–299, 1965.

[3] R. Gribonval and M. Nikolova, "A characterization of proximity operators," *J. Math. Imaging Vis.*, vol. 62, pp. 773—-789, 2020.

[4] R. Gribonval, "Should penalized least squares regression be interpreted as maximum a posteriori estimation?" *IEEE Trans. Signal Process.*, vol. 59, no. 5, pp. 2405–2410, May 2011.

[5] R. Gribonval and P. Machart, "Reconciling "priors" & "priors" without prejudice?" in *Proc. Advances in Neural Information Processing Systems 26*, Lake Tahoe, NV, USA, December 5-10, 2013, pp. 2193–2201.

[6] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, July 2017.

[7] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, "Plug-and-play image restoration with deep denoiser prior," *IEEE Trans. Patt. Anal. and Machine Intell.*, vol. 44, no. 10, pp. 6360–6376, Oct. 2022, doi: 10.1109/tpami.2021.3088914.

[8] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 123–231, 2014.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[10] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *Proc. IEEE Global Conf. Signal Process. and INf. Process. (GlobalSIP)*, Austin, TX, USA, December 3-5, 2013, pp. 945–948.

[11] S. H. Chan, X. Wang, and O. A. Elgendy, "Plug-and-play ADMM for image restoration: Fixed-point convergence and applications," *IEEE Trans. Comp. Imag.*, vol. 3, no. 1, pp. 84–98, March 2017.

[12] C. Park, S. Shoushtari, W. Gan, and U. S. Kamilov, "Convergence of nonconvex pnp-admm with mmse denoisers," in *Proc. Int. Workshop on Computational Advances in Multi-Sensor Adaptive Process. (CAMSAP 2023)*, Los Suenos, Costa Rica, Dec. 2023, pp. 511–515.

[13] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, 2004.

[14] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (RED)," *SIAM J. Imaging Sci.*, vol. 10, no. 4, pp. 1804–1844, 2017.

[15] E. T. Reehorst and P. Schniter, "Regularization by denoising: Clarifications and new interpretations," *IEEE Trans. Comput. Imag.*, vol. 5, no. 1, pp. 52–67, Mar. 2019.

[16] Y. Sun, J. Liu, and U. S. Kamilov, "Block coordinate regularization by denoising," in *Proc. Advances in Neural Information Processing Systems 32*, Vancouver, BC, Canada, Dec. 2019, pp. 382–392.

[17] S. Hurault, A. Leclaire, and N. Papadakis, "Gradient step denoiser for convergent plug-and-play," in *International Conference on Learning Representations (ICLR)*, Kigali, Rwanda, May 1-5, 2022.

[18] R. Cohen, Y. Blau, D. Freedman, and E. Rivlin, "It has potential: Gradient-driven denoisers for convergent solutions to inverse problems," in *Proc. Advances in Neural Information Processing Systems 34*, December 6-14, 2021, pp. 18 152–18 164.

[19] Y. Sun, B. Wohlberg, and U. S. Kamilov, "An online plug-and-play algorithm for regularized image reconstruction," *IEEE Trans. Comput. Imag.*, vol. 5, no. 3, pp. 395–408, Sep. 2019.

[20] Z. Wu, Y. Sun, A. Matlock, J. Liu, L. Tian, and U. S. Kamilov, "SIMBA: Scalable inversion in optical tomography using deep denoising priors," *IEEE J. Sel. Topics Signal Process.*, vol. 14, no. 6, pp. 1163–1175, 2020.

[21] Y. Sun, Z. Wu, X. Xu, B. Wohlberg, and U. S. Kamilov, "Scalable plug-and-play ADMM with convergence guarantees," *IEEE Trans. Comput. Imag.*, vol. 7, pp. 849–863, Jul. 2021.

[22] J. Liu, S. Asif, B. Wohlberg, and U. S. Kamilov, "Recovery analysis for plug-and-play priors using the restricted eigenvalue condition," in *Proc. Advances in Neural Information Processing Systems 34*, December 6-14, 2021, pp. 5921–5933.

[23] D. Gilton, G. Ongie, and R. Willett, "Deep equilibrium architectures for inverse problems in imaging," *IEEE Trans. Comput. Imag.*, vol. 7, pp. 1123–1133, Oct. 2021.

[24] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2010.

[25] ——, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, 2nd ed. Springer, 2017.