Plug-and-Play: Algorithms, Parameters Tuning and Interpretation

Zhonglin Xie

Peking University

February 28, 2022

Linear Inverse Problem

- Formulation:

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}, \quad \boldsymbol{w} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{I}\right),$$

  where $\boldsymbol{A}$ is a known block-diagonal matrix, and $\boldsymbol{w}$ denotes Gaussian random vector with mean $\mathbf{0}$ and covariance $\sigma^2\boldsymbol{I}$.
- Aim: Recovery $\boldsymbol{x}$ from $\boldsymbol{y}$.
- Application: Magnetic Resonance Imaging (MRI).

Signal Recovery and Denoising

- The maximum likelihood (ML) estimate:

$$\widehat{\boldsymbol{x}}_{\mathrm{ml}} \triangleq \underset{x}{\operatorname{argmax}}\, p(\boldsymbol{y} \mid \boldsymbol{x}),$$

  where $p(\boldsymbol{y} \mid \boldsymbol{x})$, the probability density of $\boldsymbol{y}$ conditioned on $\boldsymbol{x}$, is known as the likelihood function.

- The equivalent form:

$$\widehat{\boldsymbol{x}}_{\mathrm{ml}} = \underset{x}{\operatorname{argmin}}\{-\ln p(\boldsymbol{y} \mid \boldsymbol{x})\}.$$

- In the case of additive white Gaussian noise (AWGN) $\mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{I}\right)$, we have

$$-\ln p(\boldsymbol{y} \mid \boldsymbol{x}) = \frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \text{const}.$$

Maximum A Posteriori (MAP)

- Since $\boldsymbol{A}$ is fat, we can not perform least-squares estimation.
- Use a regularization term $\phi(\boldsymbol{x})$ to encodes priori knowledge:

$$\widehat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\{ \frac{1}{2\sigma^2} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \phi(\boldsymbol{x}) \right\}.$$

- Bayes rule implies

$$\ln p(\boldsymbol{x} \mid \boldsymbol{y}) = \ln p(\boldsymbol{y} \mid \boldsymbol{x}) + \ln p(\boldsymbol{x}) - \ln p(\boldsymbol{y}).$$

- The maximum a posteriori (MAP) estimate:

$$\widehat{\boldsymbol{x}}_{\mathrm{map}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \{-\ln p(\boldsymbol{y} \mid \boldsymbol{x}) - \ln p(\boldsymbol{x})\}.$$

- $\widehat{\boldsymbol{x}}$ can be recognized as $\widehat{\boldsymbol{x}}_{\mathrm{map}}$ with $p(\boldsymbol{x}) \propto \exp(-\phi(\boldsymbol{x}))$.

More on $\phi(\boldsymbol{x})$

- $\phi(\boldsymbol{x})$ should mimic the negative log of the prior density.
- $\phi(\boldsymbol{x})$ must enable tractable optimization.
- Common choice: $\phi(\boldsymbol{x}) = \lambda \|\Psi \boldsymbol{x}\|_1$, where $\Psi^\intercal \Psi = \boldsymbol{I}$, $\lambda > 0$.
- Advantages:
  The problem remains convex.
  The transform output $\Psi \boldsymbol{x}$ is sparse.

Denoising

- When $\boldsymbol{A} = \boldsymbol{I}$, the linear inverse problem reduces to

$$\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{w}, \quad \boldsymbol{w} \sim \mathcal{N}\left(\boldsymbol{0}, \sigma^2 \boldsymbol{I}\right).$$

  Recovering $\boldsymbol{x}$ from noisy $\boldsymbol{z}$ is known as **denoising**.
- State-of-the-art approaches are either algorithmic or neural.
- Can these state-of-the-art denoisers be leveraged for MRI?

PnP-ADMM

- $\widehat{\boldsymbol{x}}$ can be derived from an equivalent optimization

$$(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{v}}) = \underset{\boldsymbol{x},\boldsymbol{v}\in\mathbb{R}^n}{\arg\min} \ \frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \phi(\boldsymbol{v}), \qquad \text{s.t. } \boldsymbol{x} = \boldsymbol{v}.$$

- The augmented Lagrangian:

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{v}; \boldsymbol{\lambda}) &= \frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \phi(\boldsymbol{v}) - \boldsymbol{\lambda}^\mathsf{T}(\boldsymbol{x} - \boldsymbol{v}) + \frac{1}{2\eta}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2 \\ &= \frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \phi(\boldsymbol{v}) + \frac{1}{2\eta}\|\boldsymbol{x} - \boldsymbol{v} + \boldsymbol{u}\|_2^2, \end{aligned}$$

where $\boldsymbol{u} = \eta\boldsymbol{\lambda}$.

ADMM

Alternating the optimization of $\boldsymbol{x}$, $\boldsymbol{v}$ with gradient ascent of $\boldsymbol{u}$:

$$\begin{aligned}
\boldsymbol{x}_k &= \boldsymbol{h}\left(\boldsymbol{v}_{k-1} - \boldsymbol{u}_{k-1}; \sigma^2/\eta\right) \\
\boldsymbol{v}_k &= \operatorname{prox}_\phi\left(\boldsymbol{x}_k + \boldsymbol{u}_{k-1}; \eta\right) \\
\boldsymbol{u}_k &= \boldsymbol{u}_{k-1} + \left(\boldsymbol{x}_k - \boldsymbol{v}_k\right)
\end{aligned}$$

where

$$\begin{aligned}
\operatorname{prox}_\phi(\boldsymbol{z}; \eta) &\triangleq \operatorname*{arg\,min}_{\boldsymbol{x}\in\mathbb{R}^n}\ \phi(\boldsymbol{x}) + \frac{1}{2\eta}\|\boldsymbol{x} - \boldsymbol{z}\|^2, \\
\boldsymbol{h}(\boldsymbol{z}; \sigma^2/\eta) &\triangleq \operatorname*{arg\,min}_{\boldsymbol{x}\in\mathbb{R}^n}\ \frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \frac{1}{2\eta}\|\boldsymbol{x} - \boldsymbol{z}\|^2 \\
&= \left(\boldsymbol{A}^\intercal\boldsymbol{A} + \frac{\sigma^2}{\eta}\boldsymbol{I}\right)^{-1}\left(\boldsymbol{A}^\intercal\boldsymbol{y} + \frac{\sigma^2}{\eta}\boldsymbol{z}\right).
\end{aligned}$$

PnP-ADMM

- $\operatorname{prox}_{\phi}(\boldsymbol{z};\eta)$ can be **recognized** as the MAP **denoiser** of $\boldsymbol{z}$.
- PnP plug in a image denoiser in place of the $\operatorname{prox}_{\phi}(\boldsymbol{z};\eta)$.
- Denoting the denoiser as $\boldsymbol{f}(\cdot;\eta)$, PnP-ADMM writes:

$$\begin{aligned}
\boldsymbol{x}_k &= \boldsymbol{h}\left(\boldsymbol{v}_{k-1} - \boldsymbol{u}_{k-1}; \sigma^2/\eta\right) \\
\boldsymbol{v}_k &= \boldsymbol{f}\left(\boldsymbol{x}_k + \boldsymbol{u}_{k-1}; \eta\right) \\
\boldsymbol{u}_k &= \boldsymbol{u}_{k-1} + \left(\boldsymbol{x}_k - \boldsymbol{v}_k\right)
\end{aligned}$$

- The fixed point of the ADMM is independent of η, while η affects the fixed-point of the PnP-ADMM.
- To promote the PnP-ADMM, we untie the parameters:

$$\begin{aligned}
\boldsymbol{x}_k &= \boldsymbol{h}\left(\boldsymbol{v}_{k-1} - \boldsymbol{u}_{k-1}; \mu_k\right) \\
\boldsymbol{v}_k &= \boldsymbol{f}\left(\boldsymbol{x}_k + \boldsymbol{u}_{k-1}; \eta_k\right) \\
\boldsymbol{u}_k &= \boldsymbol{u}_{k-1} + \left(\boldsymbol{x}_k - \boldsymbol{v}_k\right)
\end{aligned}$$

Discussion: The Effects of the Denoising Strength η

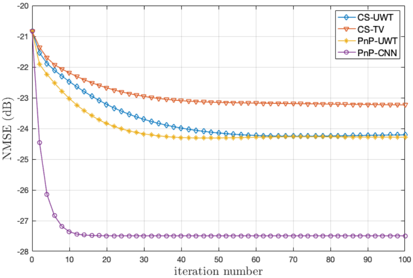
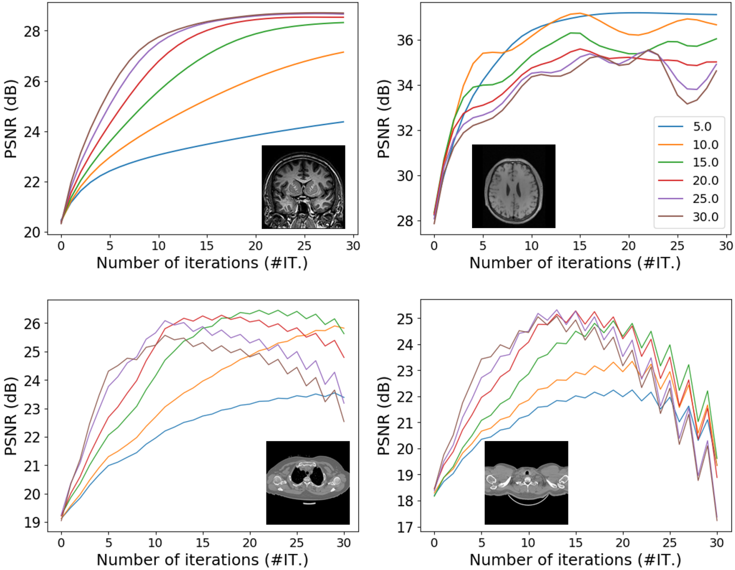


Figure: The normalized mean-squared error (NMSE) versus iteration.

- ADMM
- PnP-ADMM

The Effects of the Denoising Strength η_k



RL Formulation for Automated Parameter Selection

- Motivation: η, μ affect the result of the PnP-ADMM.
- Manually tuned parameters are time-cost.
- Aim: automatically select

  $$\tau \text{ and } (\eta_0, \mu_0, \eta_1, \mu_1, \cdots, \eta_{\tau-1}, \mu_{\tau-1})$$

  to recover $\boldsymbol{x}_\tau$ that close to $\boldsymbol{x}$.
- Tool: Reinforcement Learning (RL).

Markov decision process (MDP) $(\mathcal{S}, \mathcal{A}, p, r)$

- State space $\mathcal{S}$: any feasible value of $(\boldsymbol{x}_k, \boldsymbol{v}_k, \boldsymbol{u}_k)$.
- Action space $\mathcal{A}$: any feasible value of τ and (μ_k, η_k).
- Transition function $p\colon \mathcal{S} \times \mathcal{A} \to \mathcal{S}$

  several iterations of the PnP-ADMM.

- Reward function $r\colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$

$$r\left(s_t, a_t\right) = \left[\zeta\left(p\left(s_t, a_t\right)\right) - \zeta\left(s_t\right)\right] - \eta.$$

  $\zeta\left(s_t\right)$: the PSNR of the recovered image at step t.
  η: penalizing the policy as it does not terminate at step t.

Workflow of TFPnP

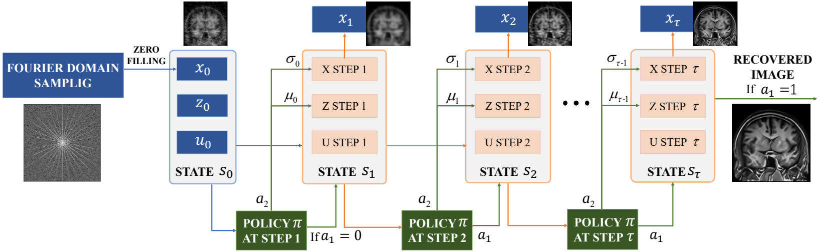


Figure: Workflow of the TFPnP instantiated by the PnP-ADMM.

Formal Definition of the Goal

- $s_k = (\boldsymbol{x}_k, \boldsymbol{v}_k, \boldsymbol{u}_k), a_k = (a_{k,1}, a_{k,2}), r_k = r(s_k, a_k)$.
- $a_{k,2} = (\mu_k, \eta_k)$. $a_{k,1}$ is a boolean that terminates the iterate at step k when $a_{k,1} = 0$ and versus verse.
- Trajectory: $T = \{s_0, a_0, r_0, \cdots, s_N, a_N, r_N\}$.
- Given T and $\rho \in [0, 1]$, define the return as

$$R_t = \sum_{t'=0}^{N-t} \rho^{t'} r\left(s_{t+t'}, a_{t+t'}\right).$$

- Goal: learn a policy π to maximize

$$J(\pi) = \mathbb{E}_{s_0,\pi}\left[R_0\right], \quad \pi(a \mid s) : \mathcal{S} \times \mathcal{A} \to [0, 1].$$

RL-based Policy Learning

- State-value function:

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[R_0 \mid s_0 = s\right]$$

- Action-value function:

$$Q^{\pi}(s, a) = \mathbb{E}_{\pi}\left[R_0 \mid s_0 = s, a_0 = a\right]$$

- Policy: $\pi = (\pi_1, \pi_2)$.
  π_1: a **stochastic** policy that generate $a_{t,1}$ to decide whether to terminate.
  π_2: a **deterministic** policy that generate $a_{t,2}$.

Actor-critic Framework

- Policy network (actor): $\pi_\theta = (\pi_1, \pi_2)$ with $\theta = (\theta_1, \theta_2)$.
  $\pi_1(\cdot \mid s)$: $\mathcal{S} \times \{0,1\} \to [0,1]$, controlled by θ_1.
  $\pi_2(s)$: $\mathcal{S} \to \mathcal{A}$, controlled by θ_2.
- Value network (critic): $V_\phi^\pi(s_t)$.
- Train the value network:

$$L_\phi = \mathbb{E}_{s\sim B, a\sim\pi_\theta(s)}\left[\frac{1}{2}\left(r(s,a) + \gamma V_{\hat{\phi}}^\pi(p(s,a)) - V_\phi^\pi(s)\right)^2\right]$$

- Model-free training of π_1:

$$\nabla_{\theta_1} J(\pi_\theta) = \mathbb{E}_{s\sim B, a\sim\pi_\theta(s)}\left[\nabla_{\theta_1} \log \pi_1(a_1 \mid s) A^\pi(s,a)\right]$$

- Model-based training of π_2:

$$\nabla_{\theta_2} J(\pi_\theta) = \mathbb{E}_{s\sim B, a\sim\pi_\theta(s)}\left[\nabla_{a_2} Q^\pi(s,a) \nabla_{\theta_2} \pi_2(s)\right]$$

Training Scheme

Algorithm 1 Training Scheme

Require: Image dataset D, degradation operator $g(\cdot)$, learning rates l_θ, l_ϕ, weight parameter β.

Initialize network parameters θ, ϕ, $\hat{\phi}$ and state buffer B.
for each training iteration **do**
    sample initial state s_0 from D via $g(\cdot)$
    for environment step $t \in [0, N)$ **do**
        $a_t \sim \pi_\theta(a_t|s_t)$
        $s_{t+1} \sim p(s_{t+1}|s_t, a_t)$
        $B \leftarrow B \cup \{s_{t+1}\}$
        break if the boolean outcome of a_t equals to 1
    end for
    for each gradient step **do**
        sample states from the state buffer B
        $\theta_1 \leftarrow \theta_1 + l_\theta \nabla_{\theta_1} J(\pi_\theta)$
        $\theta_2 \leftarrow \theta_2 + l_\theta \nabla_{\theta_2} J(\pi_\theta)$
        $\phi \leftarrow \phi - l_\phi \nabla_\phi L_\phi$
        $\hat{\phi} \leftarrow \beta\phi + (1-\beta)\hat{\phi}$
    end for
end for

Ensure: Learned policy network π_θ

Experiment Results: CS-MRI

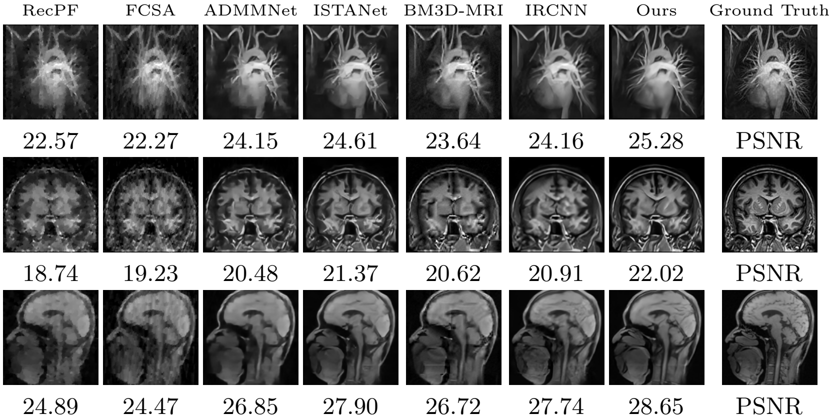


Figure: Visual and numerical CS-MRI reconstruction comparison against the state-of-the-art techniques on medical images. The numerical values denote the PSNR obtained by each technique.

Rethinking of the PnP-ADMM: On Derivation

- $\boldsymbol{f}$ is not the proximal map of any regularizer ϕ.
- $\boldsymbol{f}$ coincides with $\mathrm{prox}_\phi(\boldsymbol{z};\eta)$ **only when**

$$p(\boldsymbol{x}) \propto \exp(-\phi(\boldsymbol{x})), \quad \boldsymbol{z} - \boldsymbol{z}_{\mathrm{true}} \sim \mathcal{N}(\boldsymbol{0}, \eta^2 \boldsymbol{I}).$$

- However, $p(\boldsymbol{x})$ may **not** prompt to $\exp(-\phi(\boldsymbol{x}))$ and the distribution of

$$(\boldsymbol{x}_k + \boldsymbol{u}_{k-1}) - (\boldsymbol{x}_k + \boldsymbol{u}_{k-1})_{\mathrm{true}}$$

  is **unknown**!
- PnP-ADMM is a result of the similarity of the formulation.

Rethinking of the PnP-ADMM: On Convergence

- PnP-ADMM may not be an implementation of ADMM.
- If the PnP-ADMM converges?
- If it does converge, what it converges to?

PnP FISTA

$$\min_{\boldsymbol{x}} \; \frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \phi(\boldsymbol{x})$$

$$\begin{aligned}
\boldsymbol{z}_k &= \boldsymbol{s}_{k-1} - \frac{\eta}{\sigma^2}\boldsymbol{A}^\intercal\left(\boldsymbol{A}\boldsymbol{s}_{k-1} - \boldsymbol{y}\right) \\
\boldsymbol{x}_k &= \mathrm{prox}_\phi(\boldsymbol{z}_k; \eta) \\
\boldsymbol{s}_k &= \boldsymbol{x}_k + \frac{q_{k-1} - 1}{q_k}\left(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\right)
\end{aligned}$$

FISTA

$$\begin{aligned}
\boldsymbol{z}_k &= \boldsymbol{s}_{k-1} - \frac{\eta}{\sigma^2}\boldsymbol{A}^\intercal\left(\boldsymbol{A}\boldsymbol{s}_{k-1} - \boldsymbol{y}\right) \\
\boldsymbol{x}_k &= \boldsymbol{f}\left(\boldsymbol{z}_k\right) \\
\boldsymbol{s}_k &= \boldsymbol{x}_k + \frac{q_{k-1} - 1}{q_k}\left(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\right)
\end{aligned}$$

PnP FISTA

where it is typical to use $q_k = \left(1 + \sqrt{1 + 4q_{k-1}^2}\right)/2$ and $q_0 = 1$ with step-size $\eta \in \left(0, \sigma^2\|A\|_2^{-2}\right)$.

Regularization by Denoising (RED)

- Recover $\boldsymbol{x}$ from measurements $\boldsymbol{y}$ by solving

$$\mathbf{0} = \frac{1}{\sigma^2}\boldsymbol{A}^{\mathsf{T}}(\boldsymbol{A}\widehat{\boldsymbol{x}} - \boldsymbol{y}) + \frac{1}{\eta}(\widehat{\boldsymbol{x}} - \boldsymbol{f}(\widehat{\boldsymbol{x}})).$$

- $\boldsymbol{f}$ is an arbitrary image denoiser.
- When $\boldsymbol{f}$ is a sophisticated denoiser and η is well tuned, the solutions $\widehat{\boldsymbol{x}}$ are state-of-the-art.

RED: Assumptions

Define

$$\rho_{\text{RED}}(\boldsymbol{x}) \triangleq \frac{1}{2}\langle \boldsymbol{x}, \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})\rangle, \quad \ell(\boldsymbol{x};\boldsymbol{y}) = \frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2.$$

We get

$$\widehat{\boldsymbol{x}}_{\text{RED}} = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\arg\min}\ \ell(\boldsymbol{x};\boldsymbol{y}) + \rho_{\text{RED}}(\boldsymbol{x}).$$

The denoiser $\boldsymbol{f}(x)$ obeys the following assumption:

1. Local Homogeneity:

$$\boldsymbol{f}((1+\varepsilon)\boldsymbol{x}) = (1+\varepsilon)\boldsymbol{f}(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \mathbb{R}^n, 0 < \varepsilon \ll 1.$$

2. $\boldsymbol{f}(\cdot)$ is differentiable where $J\boldsymbol{f} \in \mathbb{R}^{n\times n}$ denotes its Jacobian.
3. Jacobian Symmetry: $J\boldsymbol{f}(\boldsymbol{x})^\intercal = J\boldsymbol{f}(\boldsymbol{x}), \forall \boldsymbol{x} \in \mathbb{R}^n$.
4. The spectral radius the Jacobian satisfies $\eta(J\boldsymbol{f}(\boldsymbol{x})) \leq 1$.

RED: Proof

- From the multivariate calculus:

$$\nabla \rho_{\text{RED}}(\boldsymbol{x}) = \boldsymbol{x} - \frac{1}{2}\boldsymbol{f}(\boldsymbol{x}) - \frac{1}{2}[J\boldsymbol{f}(\boldsymbol{x})]^{\mathsf{T}}\boldsymbol{x}.$$

- Local homogeneity implies $[J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x})$:

$$\begin{aligned} 0 &= \lim_{\varepsilon \to 0} \frac{\|\boldsymbol{f}(\boldsymbol{x} + \varepsilon\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x}) - [J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x}\varepsilon\|}{\|\varepsilon\boldsymbol{x}\|} \\ &= \lim_{\varepsilon \to 0} \frac{\|(1+\varepsilon)\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x}) - [J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x}\varepsilon\|}{\|\varepsilon\boldsymbol{x}\|} \\ &= \lim_{\varepsilon \to 0} \frac{\|\boldsymbol{f}(\boldsymbol{x}) - [J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x}\|}{\|\boldsymbol{x}\|}. \end{aligned}$$

- Jacobian symmetry gives $\nabla \rho_{\text{RED}}(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})$.
- $\eta(J\boldsymbol{f}(\boldsymbol{x})) \leq 1$ guarantees the convexity.

RED: Remark

- When the denoiser $\boldsymbol{f}(\cdot)$ is locally homogeneous, then

$$\nabla \rho_{\text{RED}}(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}) \quad \Leftrightarrow \quad J\boldsymbol{f}(\boldsymbol{x}) = [J\boldsymbol{f}(\boldsymbol{x})]^{\mathsf{T}}.$$

- When $J\boldsymbol{f}(\cdot) \neq J\boldsymbol{f}(\cdot)^{\mathsf{T}}$, there exists no regularizer $\rho(\cdot)$ for which $\nabla \rho(\boldsymbol{x}) = \boldsymbol{x} - f(\boldsymbol{x})$.
- Many popular denoisers lack symmetric Jacobian, making the gradient expression invalid.

Proximal-based PnP v.s. RED: a Toy Example

- $\boldsymbol{f}(\boldsymbol{z}) = \boldsymbol{W}\boldsymbol{z}$ with $\boldsymbol{W} = \boldsymbol{W}^\top$.
- $\boldsymbol{f}$ is the proximal map of $\phi(\boldsymbol{x}) = (1/2\eta)\boldsymbol{x}^\top \left(\boldsymbol{W}^{-1} - \boldsymbol{I}\right)\boldsymbol{x}$.
- Proximal-based PnP:

$$\widehat{\boldsymbol{x}}_{\text{pnp}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\{ \frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \frac{1}{2\eta}\boldsymbol{x}^\top \left(\boldsymbol{W}^{-1} - \boldsymbol{I}\right)\boldsymbol{x} \right\}$$

- RED:

$$\widehat{\boldsymbol{x}}_{\text{red}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\{ \frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \frac{1}{2\eta}\boldsymbol{x}^\top (\boldsymbol{I} - \boldsymbol{W})\boldsymbol{x} \right\}$$

Algorithms for RED

- GD, inexact ADMM, and a "fixed-point" heuristic that was later recognized as a special case of the proximal gradient (PG) algorithm.
- Accelerated proximal gradient (fastest):

$$
\begin{aligned}
\boldsymbol{x}_k &= \boldsymbol{h}\left(\boldsymbol{v}_{k-1}; \eta/L\right) \\
\boldsymbol{z}_k &= \boldsymbol{x}_k + \frac{q_{k-1}-1}{q_k}\left(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\right) \\
\boldsymbol{v}_k &= \frac{1}{L}\boldsymbol{f}\left(\boldsymbol{z}_k\right) + \left(1 - \frac{1}{L}\right)\boldsymbol{z}_k
\end{aligned}
$$

where $L > 0$ is a design parameter that can be related to the Lipschitz constant of $\phi_{\text{red}}(\cdot)$.

RED as Score Matching

- Given a training set $\{\boldsymbol{x}_t\}_{t=1}^{T}$, the empirical prior model is

$$\widehat{p}(\boldsymbol{x}) \triangleq \frac{1}{T}\sum_{t=1}^{T}\delta\left(\boldsymbol{x}-\boldsymbol{x}_t\right)$$

- Build a prior model using kernel density estimation (KDE):

$$\tilde{p}(\boldsymbol{x};\eta) \triangleq \frac{1}{T}\sum_{t=1}^{T}\mathcal{N}\left(\boldsymbol{x};\boldsymbol{x}_t,\eta\boldsymbol{I}\right)$$

- Adopting $\tilde{p}$ as the prior, MAP becomes

$$\widehat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}}\ \frac{1}{2\sigma^2}\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|^2 - \ln\tilde{p}(\boldsymbol{x};\eta)$$

RED as Score Matching

- Because $\ln \tilde{p}$ is differentiable, $\widehat{\boldsymbol{x}}$ must obey

$$\mathbf{0} = \frac{1}{\sigma^2}\boldsymbol{A}^\top(\boldsymbol{A}\widehat{\boldsymbol{x}} - \boldsymbol{y}) - \nabla \ln \tilde{p}(\widehat{\boldsymbol{x}}; \eta)$$

- $\boldsymbol{f}_{\text{mmse}}(\boldsymbol{z}; \eta) = \mathbb{E}[\boldsymbol{x}|\boldsymbol{z}]$, where $\boldsymbol{z} = \boldsymbol{x} + \mathcal{N}(\mathbf{0}, \eta\boldsymbol{I}), \boldsymbol{x} \sim \widehat{p}$
- Tweedie's formula says that

$$\nabla \ln \tilde{p}(\boldsymbol{z}; \eta) = \frac{1}{\eta}\left(\boldsymbol{f}_{\text{mmse}}(\boldsymbol{z}; \eta) - \boldsymbol{z}\right)$$

- The MAP estimate $\widehat{\boldsymbol{x}}$ under the KDE prior $\tilde{p}$ obeys

$$\mathbf{0} = \frac{1}{\sigma^2}\boldsymbol{A}^\top(\boldsymbol{A}\widehat{\boldsymbol{x}} - \boldsymbol{y}) + \frac{1}{\eta}\left(\widehat{\boldsymbol{x}} - \boldsymbol{f}_{\text{mmse}}(\widehat{\boldsymbol{x}}; \eta)\right)$$

  which matches the RED condition when $\boldsymbol{f} = \boldsymbol{f}_{\text{mmse}}(\cdot; \eta)$

RED as Score Matching: $\boldsymbol{f} \neq \boldsymbol{f}_{\text{mmse}}(\cdot;\eta)$

- $\boldsymbol{f}_\theta$: neural denoiser parameterized by θ
- Training strategy:

$$\min_\theta \mathbb{E}\|\boldsymbol{x} - \boldsymbol{f}_\theta(\boldsymbol{z})\|^2, \quad \text{where} \quad \boldsymbol{x} \sim \widehat{p}, \quad \boldsymbol{z} = \boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, \eta \boldsymbol{I})$$

- MMSE orthogonality principle:

$$\begin{aligned}\mathbb{E}\left\|\boldsymbol{x} - \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{z})\right\|^2 =& \mathbb{E}\left\|\boldsymbol{x} - \boldsymbol{f}_{\text{mmse}}(z;\eta)\right\|^2 \\ &+ \mathbb{E}\left\|\boldsymbol{f}_{\text{mmse}}(\boldsymbol{z};\eta) - \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{z})\right\|^2\end{aligned}$$

- Using Tweedie's formula, we get

$$\begin{aligned}\widehat{\boldsymbol{\theta}} &= \operatorname*{argmin}_{\boldsymbol{\theta}} \mathbb{E}\|\boldsymbol{x} - \boldsymbol{f}_\theta(\boldsymbol{z})\|^2 \\ &= \operatorname*{argmin}_{\boldsymbol{\theta}} \mathbb{E}\left\|\boldsymbol{f}_{\text{mmse}}(z;\eta) - \boldsymbol{f}_{\boldsymbol{\theta}}(z)\right\|^2 \\ &= \operatorname*{argmin}_{\boldsymbol{\theta}} \mathbb{E}\|\nabla \ln \tilde{p}(z;\eta) - \frac{1}{\eta}\left(f_{\boldsymbol{\theta}}(z) - z\right)\|^2\end{aligned}$$

- Choose $\boldsymbol{\theta}$ so that $(f_\theta(z) - z)/\eta$ matches the "score" $\nabla \ln \tilde{p}$

CE for Prox-based PnP

- View Prox-based PnP as seeking a solution to

$$\widehat{\boldsymbol{x}}_{\text{pnp}} = \boldsymbol{h}\left(\widehat{\boldsymbol{x}}_{\text{pnp}} - \widehat{\boldsymbol{u}}_{\text{pnp}}; \eta\right)$$
$$\widehat{\boldsymbol{x}}_{\text{pnp}} = \boldsymbol{f}\left(\widehat{\boldsymbol{x}}_{\text{pnp}} + \widehat{\boldsymbol{u}}_{\text{pnp}}\right)$$

- It equals to find a fixed point of

$$\underline{z} = (2\boldsymbol{G} - \boldsymbol{I})(2\mathcal{F} - \boldsymbol{I})\underline{z}$$
$$\underline{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}, \quad \mathcal{F}(\underline{z}) = \begin{bmatrix} \boldsymbol{h}\left(z_1; \eta\right) \\ \boldsymbol{f}\left(z_2\right) \end{bmatrix}, \quad \mathcal{G}(\underline{z}) = \begin{bmatrix} \left(z_1 + z_2\right)/2 \\ \left(z_1 + z_2\right)/2 \end{bmatrix}$$

- Mann iteration writes:

$$\underline{\boldsymbol{z}}^{(k+1)} = (1-\gamma)\underline{\boldsymbol{z}}^{k} + \gamma(2\boldsymbol{G} - \boldsymbol{I})(2\mathcal{F} - \boldsymbol{I})\underline{\boldsymbol{z}}^{(k)}$$

CE for RED

- CE for ADMM-based RED:

$$\widehat{\boldsymbol{x}}_{\text{red}} = \boldsymbol{h}\left(\widehat{\boldsymbol{x}}_{\text{red}} - \widehat{\boldsymbol{u}}_{\text{red}}; \eta\right)$$

$$\widehat{\boldsymbol{x}}_{\text{red}} = \left(\left(1 + \frac{1}{L}\right)\boldsymbol{I} - \frac{1}{L}\boldsymbol{f}\right)^{-1}\left(\widehat{\boldsymbol{x}}_{\text{red}} + \widehat{\boldsymbol{u}}_{\text{red}}\right)$$

- A more intuitive form:

$$\widehat{\boldsymbol{x}}_{\text{red}} = \boldsymbol{h}\left(\widehat{\boldsymbol{x}}_{\text{red}} - \widehat{\boldsymbol{u}}_{\text{red}}; \eta\right)$$

$$\widehat{\boldsymbol{x}}_{\text{red}} = \boldsymbol{f}\left(\widehat{\boldsymbol{x}}_{\text{red}}\right) + L\widehat{\boldsymbol{u}}_{\text{red}}$$

- Solving the first equation gives:

$$\widehat{\boldsymbol{u}}_{\text{red}} = \frac{\eta}{\sigma^2}\boldsymbol{A}^{\mathsf{T}}\left(\boldsymbol{y} - \boldsymbol{A}\widehat{\boldsymbol{x}}_{\text{red}}\right)$$

- Plugging $\widehat{\boldsymbol{u}}_{\text{red}}$ back:

$$\frac{L\eta}{\sigma^2}A^{\mathsf{T}}\left(A\widehat{\boldsymbol{x}}_{\text{red}} - y\right) = f\left(\widehat{\boldsymbol{x}}_{\text{red}}\right) - \widehat{\boldsymbol{x}}_{\text{red}}$$

RED via Fixed-point Projection (RED-PRO)

RED-PRO problem writes:

$$\hat{\boldsymbol{x}}_{\mathrm{RED-PRO}} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \ell(\boldsymbol{x};\boldsymbol{y}), \qquad \text{s.t. } \boldsymbol{x} \in \mathrm{Fix}(\boldsymbol{f}).$$

- Interpretation: searching for a minimizer of $\ell(\boldsymbol{x};\boldsymbol{y})$ over the set of "clean" images.
- The manifold of natural images $\mathcal{M}$ is generally not well-defined, it is not easy accessible and it is not convex, making the search within this domain difficult. Therefore, as an alternative, we propose to use $\mathrm{Fix}(f)$ which is well-behaved for demicontractive denoisers and should satisfy $\mathcal{M} \subset \mathrm{Fix}(f)$ for a "perfect" denoiser.
- Common denoisers are far from being ideal, hence, the solution is sensitive to the choice of the denoiser and it may vary considerably for different choices.

d-demicontractive Mapping

A mapping T is d-demicontractive ($d \in [0, 1)$) if for any $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{z} \in \mathrm{Fix}(T)$ it holds that

$$\|T(\boldsymbol{x}) - \boldsymbol{z}\|^2 \leq \|\boldsymbol{x} - \boldsymbol{z}\|^2 + d\|T(\boldsymbol{x}) - \boldsymbol{x}\|^2$$

or equivalently

$$\frac{1-d}{2}\|\boldsymbol{x} - T(\boldsymbol{x})\|^2 \leq \langle \boldsymbol{x} - T(\boldsymbol{x}), \boldsymbol{x} - \boldsymbol{z} \rangle$$

RED-PRO

- Assume the denoiser $\boldsymbol{f}(\cdot)$ is a d-demicontractive mapping. Then, RED-PRO defines a convex minimization problem.
- Consider a demicontractive denoiser $\boldsymbol{f}(\cdot)$ and assume $\boldsymbol{f}(0) = 0$. Then,

$$\rho_{\text{RED}}(\boldsymbol{x}) = \frac{1}{2}\langle \boldsymbol{x}, \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})\rangle = 0 \text{ iff } \boldsymbol{x} \in \text{Fix}(\boldsymbol{f}).$$

- Hybrid steepest descent method for RED-PRO:

$$\begin{aligned} \boldsymbol{v}_{k+1} &= \boldsymbol{x}_k - \mu_k \nabla \ell\left(\boldsymbol{x}_k; \boldsymbol{y}\right), \\ \boldsymbol{z}_{k+1} &= f\left(\boldsymbol{v}_{k+1}\right), \\ \boldsymbol{x}_{k+1} &= (1-\alpha)\boldsymbol{v}_{k+1} + \alpha \boldsymbol{z}_{k+1}, \end{aligned}$$

- which is equivalent to

$$\boldsymbol{x}_{k+1} = f_\alpha(\boldsymbol{x}_k - \mu_k \nabla \ell(\boldsymbol{x}_k; \boldsymbol{y})), \text{ where } f_\alpha = (1-\alpha)\text{Id} + \alpha f.$$

Uniform Algorithm Framework

- accelerated-PG (proximal gradient) RED algorithm, which uses the iterative update:

$$
\begin{aligned}
\boldsymbol{v}_{k+1} &= \boldsymbol{x}_k - \mu_k \nabla \ell\left(\boldsymbol{x}_k; \boldsymbol{y}\right), \\
\boldsymbol{z}_{k+1} &= \boldsymbol{v}_{k+1} + q_k\left(\boldsymbol{v}_{k+1} - \boldsymbol{v}_k\right), \ \text{(FISTA-like acceleration)} \\
\boldsymbol{x}_{k+1} &= (1-\alpha)\boldsymbol{z}_{k+1} + \alpha f\left(\boldsymbol{z}_{k+1}\right), \ \text{(SOR-like acceleration)}
\end{aligned}
$$

- Thus, when we set $q_k \equiv 0$, i.e. when we skip the acceleration step, the above RED variant reduces to the iterative update of the Hybrid steepest for RED-PRO.
- When we continue and set $\alpha = 1$, we obtain the PnP-PGD method (Proximal-based).

Projection Gradient Descent

- Projected Gradient Descent writes

$$\boldsymbol{x}_{k+1} = P_{\mathrm{Fix}(f)}\left(\boldsymbol{x}_k - \mu_k \nabla \ell(\boldsymbol{x}; y)\right)$$

- Replacing the projection operator $P_{\mathrm{Fix}(f)}(\cdot)$ with denoiser (Plug and Play) $f(\cdot)$ we get PnP-PGD:

$$\boldsymbol{x}_{k+1} = f\left(\boldsymbol{x}_k - \mu_k \nabla \ell(\boldsymbol{x}; y)\right)$$

Convergence Theorem

Let $f(\cdot)$ be a continuous d-demicontractive denoiser and $\ell(\cdot; \boldsymbol{y})$ be a proper convex lower semicontinuous differentiable function with L-Lipschitz gradient $\nabla \ell(\cdot; \boldsymbol{y})$. Assume the following:

(A1) $\alpha \in (0, \frac{1-d}{2})$.

(A2) $\{\mu_k\}_{k \in \mathbb{N}} \subset [0, \infty)$ where $\mu_k \underset{k \to \infty}{\to} 0$ and $\sum_{k \in \mathbb{N}} \mu_k = \infty$.

Then, the sequence $\{\boldsymbol{x}_k\}_{k \in \mathbb{N}}$ generated by

$$\boldsymbol{x}_{k+1} = f_\alpha(\boldsymbol{x}_k - \mu_k \nabla \ell(\boldsymbol{x}_k; \boldsymbol{y})), \text{ where } f_\alpha = (1-\alpha)\mathrm{Id} + \alpha f,$$

converges to an optimal solution of the RED-PRO problem:

$$\hat{\boldsymbol{x}}_{\mathrm{RED-PRO}} = \arg\min_{\boldsymbol{x} \in \mathbb{R}^n} \ell(\boldsymbol{x}; \boldsymbol{y}), \qquad \text{s.t. } \boldsymbol{x} \in \mathrm{Fix}(f).$$

Conclusion

There are various ways to model the denoising problem:

1. PnP: Inspired by ADMM, Proximal gradient, while lacking objective function.
2. RED: Regularization by Denoising, while many denoisers do not satisfy the assumptions.
3. RED-PRO: require the denoisers to be demicontractive.

However, as pointed by, when applying practical algorithms (e.g. PnP-ADMM and PnP primal-dual hybrid gradient method (PnP-PDHG) , satisfy the same fixed-point equation as PnP-PGM (Proximal Gradient Method)) to solve these models, different models have the same aim:

$$\boldsymbol{x}_* = f_\alpha(\boldsymbol{x}_* - \mu_k \nabla \ell(\boldsymbol{x}_*; \boldsymbol{y})), \text{ where } f_\alpha = (1-\alpha)\mathrm{Id} + \alpha f.$$

Thus, we only need to guarantee the convergence of the above formulation.

Future Directions

- RL for general parameters tuning
- The convergence theory of the PnP with weaker assumptions
- PnP for general ADMM-based algorithms