Tuning-free Plug-and-Play Proximal Algorithm for Inverse Imaging Problems

Kaixuan Wei[1], Angelica Aviles-Rivero[2], Jingwei Liang[2],
Ying Fu[1], Carola-Bibiane Schnlieb[2], Hua Huang[1]

[1]Beijing Institute of Technology

[2]University of Cambridge

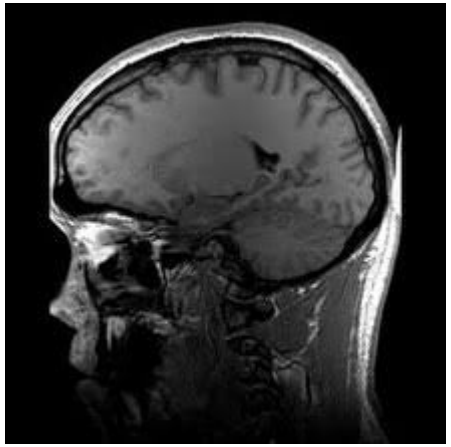
Outline

- Introduction of Plug-and-Play (PnP)
- Our approach
- Experiments on both CS-MRI and Phase-retrieval
- Conclusion

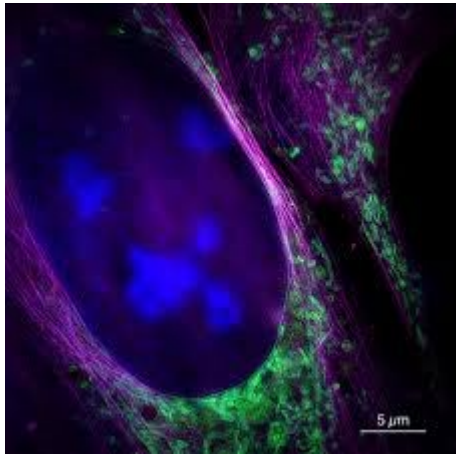
Outline

- Introduction of Plug-and-Play (PnP)
- Our approach
- Experiments on both CS-MRI and Phase-retrieval
- Conclusion

Image recovery



MRI and CT



Microscopy

Black hole imaging

Image recovery via optimization

The image recovery task is often formulated as:

$$\underset{x \in \mathbb{R}^N}{\text{minimize}} \quad \mathcal{D}(x) + \lambda \mathcal{R}(x),$$

- x is the underlying unknown image
- D is a data-fidelity term
- R is a regularizer

Image recovery via optimization

The image recovery task is often formulated as:

$$\underset{x \in \mathbb{R}^N}{\text{minimize}} \quad \mathcal{D}(x) + \lambda \mathcal{R}(x),$$

The problem is often solved by first-order iterative proximal algorithms, e.g., FISTA (Beck & Teboulle, 2009) and ADMM (Boyd et al., 2011)

Proximal operator and ADMM

To handle the non-smoothness, first-order algorithms rely on the proximal operators defined by

$$\mathrm{Prox}_{\sigma^2\mathcal{R}}(v) = \underset{x}{\mathrm{argmin}} \left(\mathcal{R}(x) + \frac{1}{2\sigma^2}\|x - v\|_2^2\right).$$

ADMM can be written as

$$\begin{aligned}
x_{k+1} &= \mathrm{Prox}_{\sigma_k^2\mathcal{R}}\left(z_k - u_k\right), \\
z_{k+1} &= \mathrm{Prox}_{\frac{1}{\mu_k}\mathcal{D}}\left(x_{k+1} + u_k\right), \\
u_{k+1} &= u_k + x_{k+1} - z_{k+1},
\end{aligned}$$

Plug-and-Play prior

Given the mathematical equivalence, the proximal operators $Prox_{\sigma^2 R}$ can be replaced by any off-the-shelf denoiser H_σ

This yields a new framework namely plug-and-play (PnP) prior (Venkatakrishnan et al., 2013), i.e.,

$$
\begin{aligned}
x_{k+1} &= \mathrm{Prox}_{\sigma_k^2 \mathcal{R}}\left(z_k - u_k\right) = \mathcal{H}_{\sigma_k}\left(z_k - u_k\right), \\
z_{k+1} &= \mathrm{Prox}_{\frac{1}{\mu_k}\mathcal{D}}\left(x_{k+1} + u_k\right), \\
u_{k+1} &= u_k + x_{k+1} - z_{k+1},
\end{aligned}
$$

Plug-and-Play prior

$$x_{k+1} = \mathrm{Prox}_{\sigma_k^2 \mathcal{R}}\left(z_k - u_k\right) = \mathcal{H}_{\sigma_k}\left(z_k - u_k\right),$$

- Denoiser priors
  - BM3D (Heide et al., 2014)
  - Nonlocal means (Venkatakrishnan et al., 2013)
  - Deep learning-based denoisers (Meinhardt et al., 2017)

  …

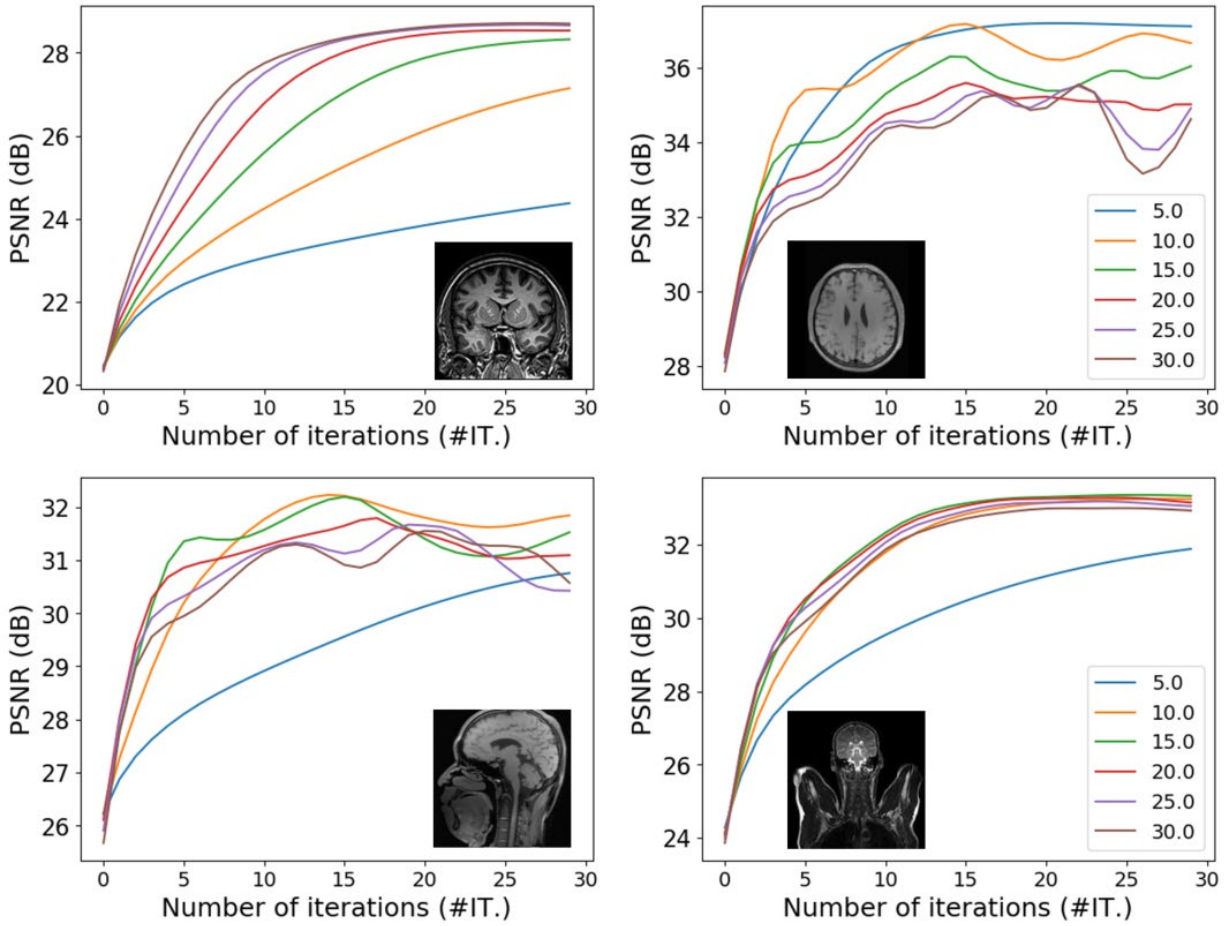
Key challenges of Plug-and-Play prior

- highly sensitive to the internal parameter selection, which generically includes the penalty parameter μ, the denoising strength σ and the terminal time τ

- Manual parameter tweaking requires several trials, which is very cumbersome and time-consuming …

Key challenges of Plug-and-Play prior

The optimal parameter setting differs image-by-image



Compressed Sensing MRI using radial sampling pattern, where PSNR curves of four medical images are displayed - using PnP-ADMM with different denoising strengths.

Our contributions

- We present **a tuning-free (TF) PnP algorithm** that can customize parameters towards diverse images

- We introduce an efficient **mixed model-free and model-based reinforcement learning (RL) algorithm** that can optimize terminal time and denoising strength/penalty parameters jointly

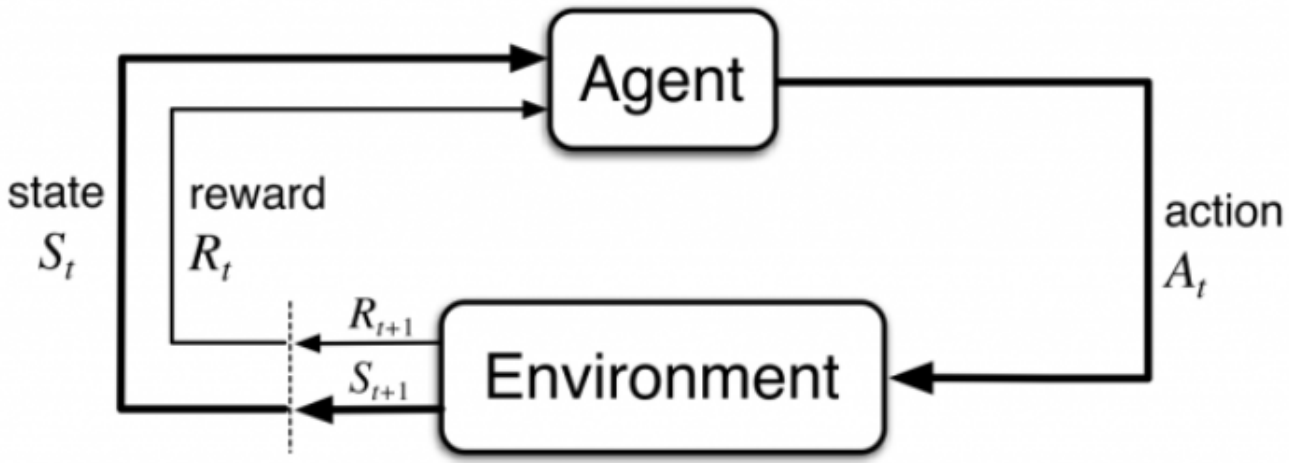
- We show that our well-designed approach leads to better results than state-of-the-art techniques on compressed sensing MRI and phase retrieval

Outline

- Introduction of Plug-and-Play (PnP)
- Our approach
- Experiments on both CS-MRI and Phase-retrieval
- Conclusion

RL formulation for automated parameter selection

We aim to select a sequence of parameters $(\sigma_0, \mu_0, \sigma_1, \mu_1, \cdots, \sigma_{\tau-1}, \mu_{\tau-1})$ to guide optimization



Markov decision process

PnP-ADMM

$$x_{k+1} = \mathcal{H}_{\sigma_k}(z_k - u_k),$$

$$z_{k+1} = \mathrm{Prox}_{\frac{1}{\mu_k}\mathcal{D}}(x_{k+1} + u_k),$$

$$u_{k+1} = u_k + x_{k+1} - z_{k+1},$$

RL formulation for automated parameter selection

We denote the MDP by the tuple $(\mathcal{S}, \mathcal{A}, p, r)$,

- S is the space of optimization variable states (x_k, z_k, u_k)
- A is the space of internal parameters (τ, σ_k, μ_k)
- P is the transition function $p : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$
- The state transition can be expressed as $s_{t+1} = p(s_t, a_t)$
- r is the reward function $r : S \times \mathcal{A} \mapsto \mathbb{R}$

RL formulation for automated parameter selection

Applying a sequence of parameters to the initial state s_0 results in a trajectory T of states, action and rewards:

$$T = \{s_0, a_0, r_0, \cdots, s_N, a_N, r_N\}$$

Given a trajectory T, we define the return r_t^γ as the summation of discounted rewards after s_t,

$$r_t^\gamma = \sum_{t'=0}^{N-t} \gamma^{t'} r(s_{t+t'}, a_{t+t'}),$$

RL formulation for automated parameter selection

Our goal is to learn a policy π, denoted as $\pi(a|s) : \mathcal{S} \mapsto \mathcal{A}$ for the decision-making agent, to maximize the objective defined as

$$J(\pi) = \mathbb{E}_{s_0 \sim S_0, T \sim \pi}\left[r_0^{\gamma}\right],$$

The expected return on states and state-action pairs under the policy π are defined by state-value functions V^{π} and action-value function Q^{π}, i.e.,

$$V^{\pi}(s) = \mathbb{E}_{T \sim \pi}\left[r_0^{\gamma} | s_0 = s\right],$$
$$Q^{\pi}(s, a) = \mathbb{E}_{T \sim \pi}\left[r_0^{\gamma} | s_0 = s, a_0 = a\right].$$

RL formulation for automated parameter selection

We decompose actions into two parts: a discrete decision a_1 on terminal time and a continuous decision a_2 on denoising strength and penalty parameter:

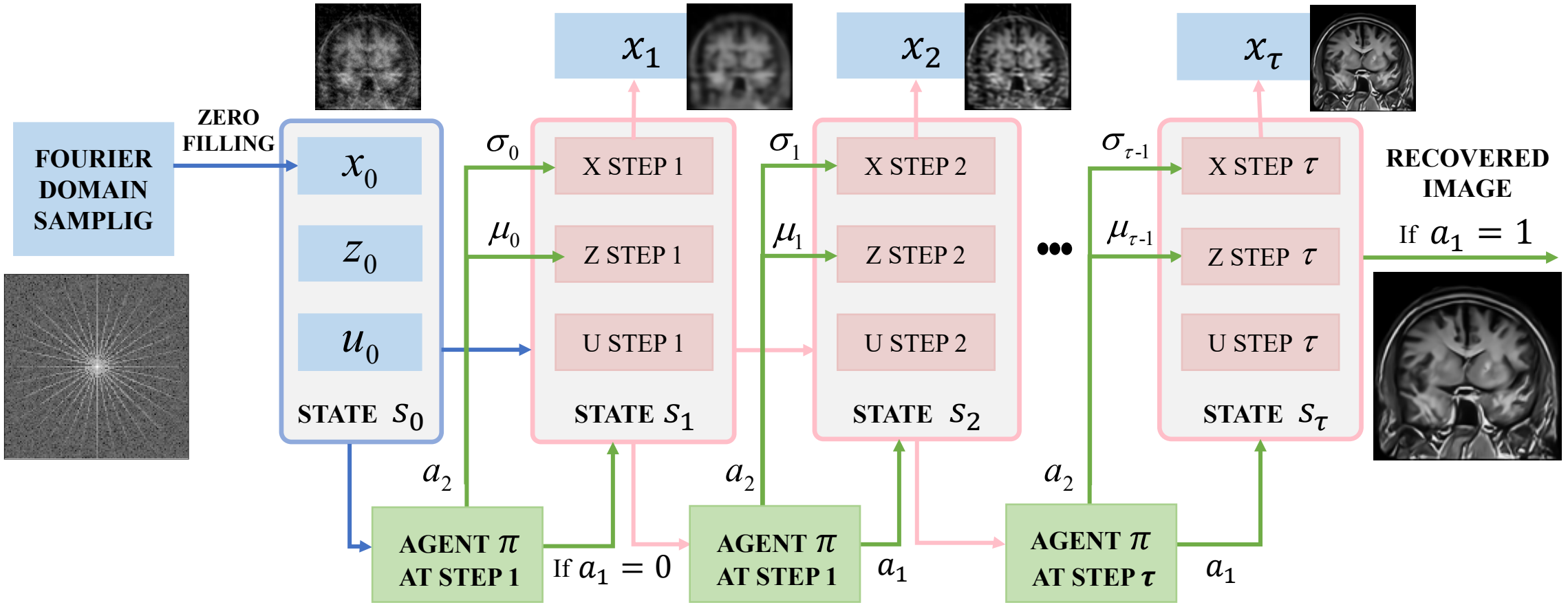
$$a = (a_1, a_2)$$

The policy also consists of two sub-polices:

$$\pi = (\pi_1, \pi_2)$$

a stochastic policy and a deterministic policy that generate a_1 and a_2 respectively

Overview of our TF-PnP approach



Environment Model

Transition function p: To reduce the computation cost, we define the transition function p to involve m iterations of the optimization.

Reward function r: To take both image recovery performance and runtime efficiency into account, we define the reward function as

$$r(s_t, a_t) = \zeta(p(s_t, a_t)) - \zeta(s_t) - \eta.$$

$\zeta(s_t)$ denotes the PSNR of the recovered image at step t

RL-based policy learning

We apply the actor-critic framework (Sutton et al., 2000) that use a policy network $\pi_\theta(a_t|s_t)$ and a value network $V_\phi^\pi(s_t)$

The network configuration

LAYER NAME	FEATURE EXTRACTOR	OUTPUT SIZE
conv1	$3 \times 3, 64$	64×64
layer1	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$	32×32
layer2	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$	16×16
layer3	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$	8×8
layer4	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$	4×4
avgpool	4×4 average pool	1×1

RL-based policy learning

The policy network is trained using policy gradient methods (Peters & Schaal, 2006)

We apply the policy gradient theorem (Sutton et al., 2000) for π_1

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

deterministic policy gradient theorem (Silver et al., 2014) for π_2

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

Outline

- Introduction of Plug-and-Play (PnP)
- Our approach
- Experiments on both CS-MRI and Phase-retrieval
- Conclusion

Compressed Sensing MRI

The forward model of CS-MRI can be mathematically described as

$$y = \mathcal{F}_p x + \omega,$$

The data-fidelity term is

$$\mathcal{D}(x) = \tfrac{1}{2}\|y - \mathcal{F}_p x\|^2$$

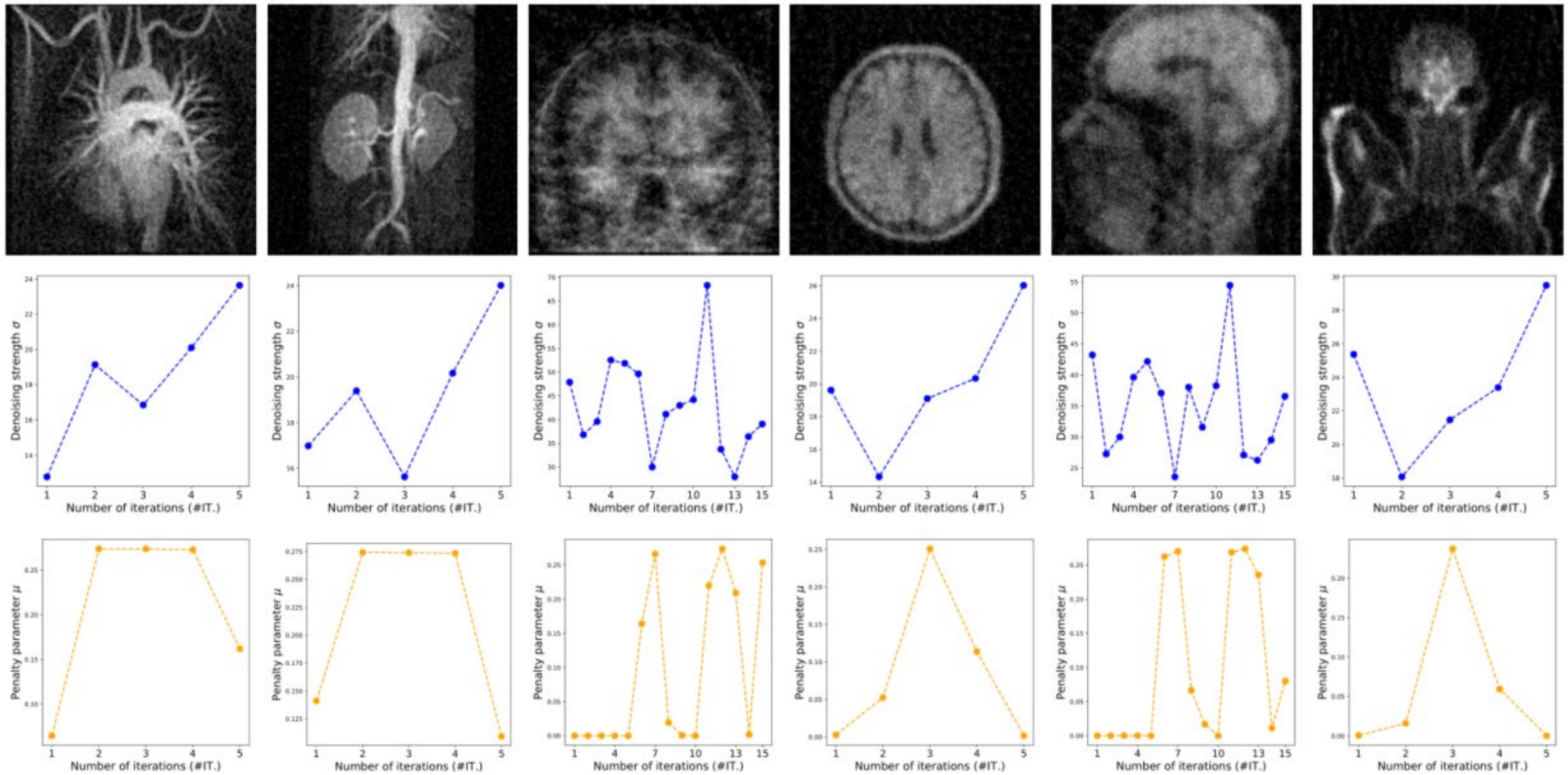
whose proximal operator is given in (Eksioglu, 2016).

Comparisons of different policies

Policies	×2		×4		×8	
	PSNR	#IT.	PSNR	#IT.	PSNR	#IT.
handcrafted	30.05	30.0	27.90	30.0	25.76	30.0
handcrafted*	30.06	29.1	28.20	18.4	26.06	19.4
fixed	23.94	30.0	24.26	30.0	22.78	30.0
fixed*	28.45	1.6	26.67	3.4	24.19	7.3
fixed optimal	30.02	30.0	28.27	30.0	26.08	16.7
fixed optimal*	30.03	6.7	28.34	12.6	26.16	30.0
oracle	30.25	30.0	28.60	30.0	26.41	30.0
oracle*	30.26	8.0	28.61	13.9	26.45	21.6
model-free	28.79	30.0	27.95	30.0	26.15	30.0
Ours	30.33	5.0	28.42	5.0	26.44	15.0

We show both PSNR and the number of iterations (#IT.) used to induce results. * denotes to report the best PSNR over all iterations

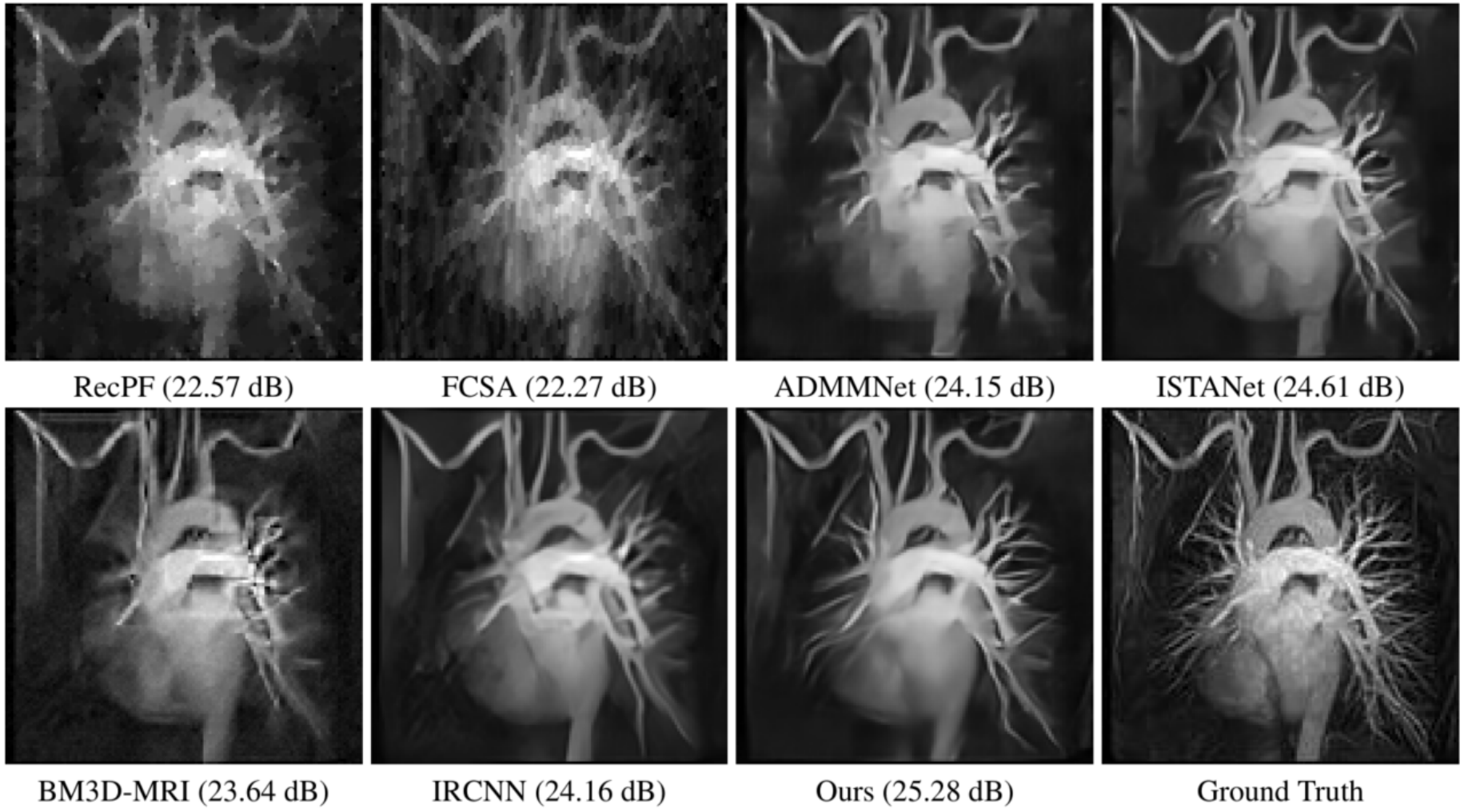
Behaviors of our learned policy



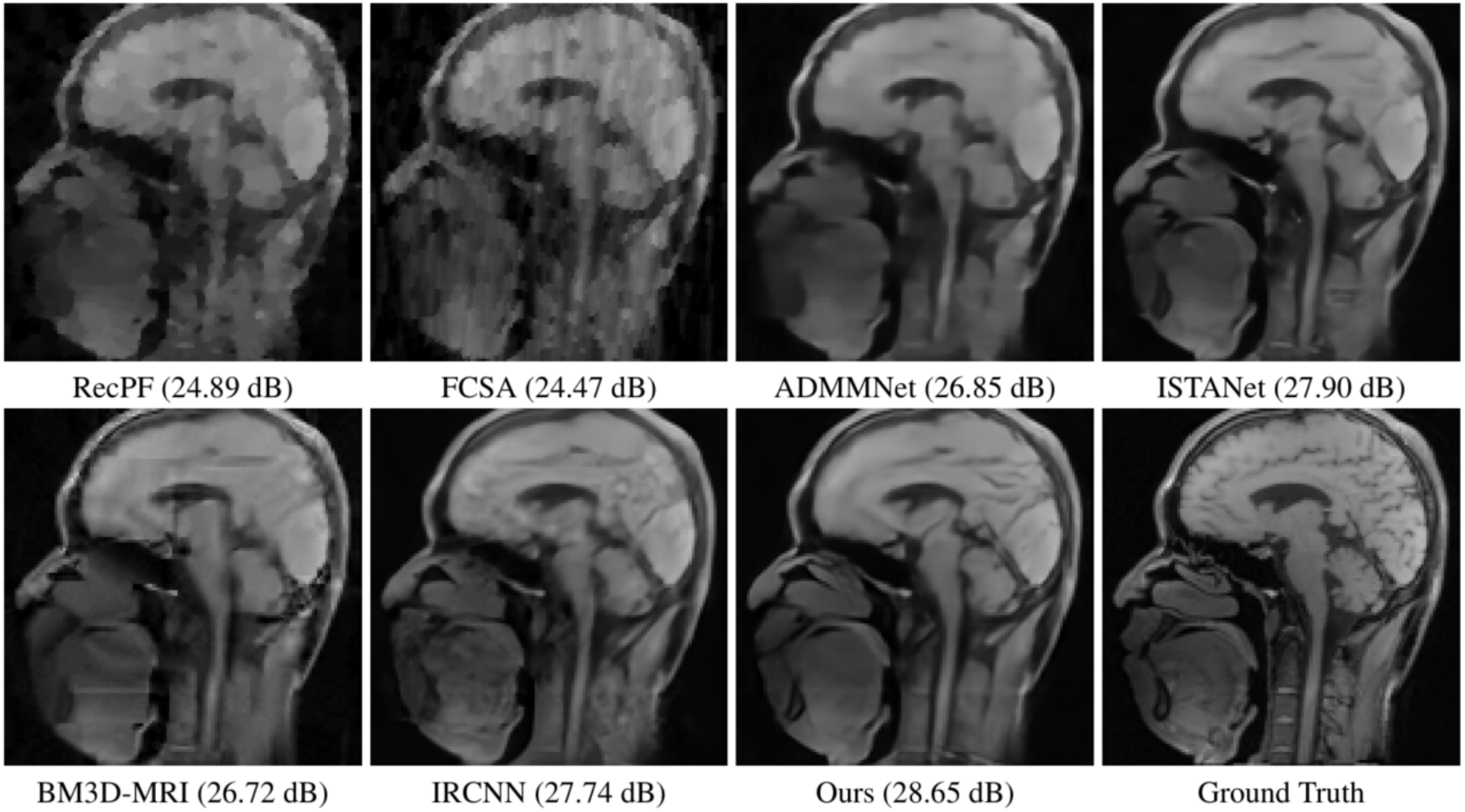
Comparisons with state-of-the-arts

Dataset	f	σ_n	Traditional		Deep Unrolling		PnP		
			RecPF	FCSA	ADMMNet	ISTANet	BM3D-MRI	IRCNN	Ours
Medical7	×2	5	32.46	31.70	33.10	34.58	33.33	34.67	34.78
		10	29.48	28.33	31.37	31.81	29.44	31.80	32.00
		15	27.08	25.52	29.16	29.99	26.90	29.96	30.27
	×4	5	28.67	28.21	30.24	31.34	30.33	31.36	31.62
		10	26.98	26.67	29.20	29.71	28.30	29.52	29.68
		15	25.58	24.93	27.87	28.38	26.66	27.94	28.43
	×8	5	24.72	24.62	26.57	27.65	26.53	27.32	28.26
		10	23.94	24.04	26.21	26.90	25.81	26.44	27.35
		15	23.18	23.36	25.49	26.23	25.09	25.53	26.41
MICCAI	×2	5	36.39	34.90	36.74	38.17	36.00	38.42	38.57
		10	31.95	30.12	34.20	34.81	31.39	34.93	35.06
		15	28.91	26.68	31.42	32.65	28.46	32.81	33.09
	×4	5	33.05	32.30	34.15	35.46	34.79	35.80	36.11
		10	30.21	29.56	32.58	33.13	31.63	32.99	33.07
		15	28.13	26.93	30.55	31.48	29.35	30.98	31.42
	×8	5	28.35	28.71	30.36	31.62	31.34	31.66	32.64
		10	26.86	27.68	29.78	30.54	29.86	30.16	30.89
		15	25.70	26.35	28.83	29.50	28.53	28.72	29.65

Visual results



Visual results



Phase retrieval

Mathematically, PR refers to the problem of recovering a vectorized signal x from measurements y of the form

$$y = |\mathbf{A}x| + w,$$

used Plug-and-Play ADMM to estimate x by solving the optimization problem

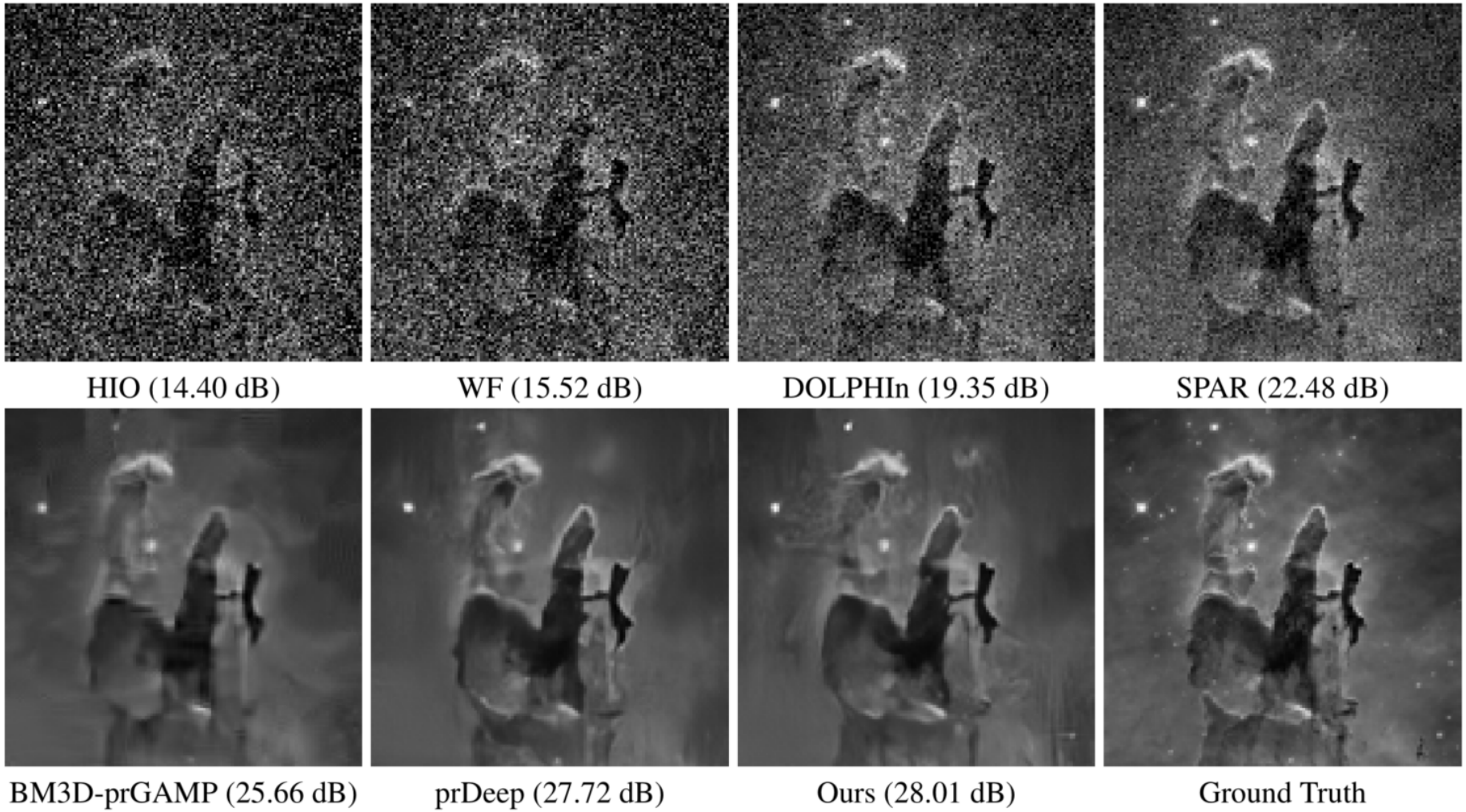
$$\arg\min_x \underbrace{\frac{1}{2}\|y - |\mathbf{A}x|\|_2^2}_{\mathcal{D}(x)} + R(x),$$

Comparisons with state-of-the-arts

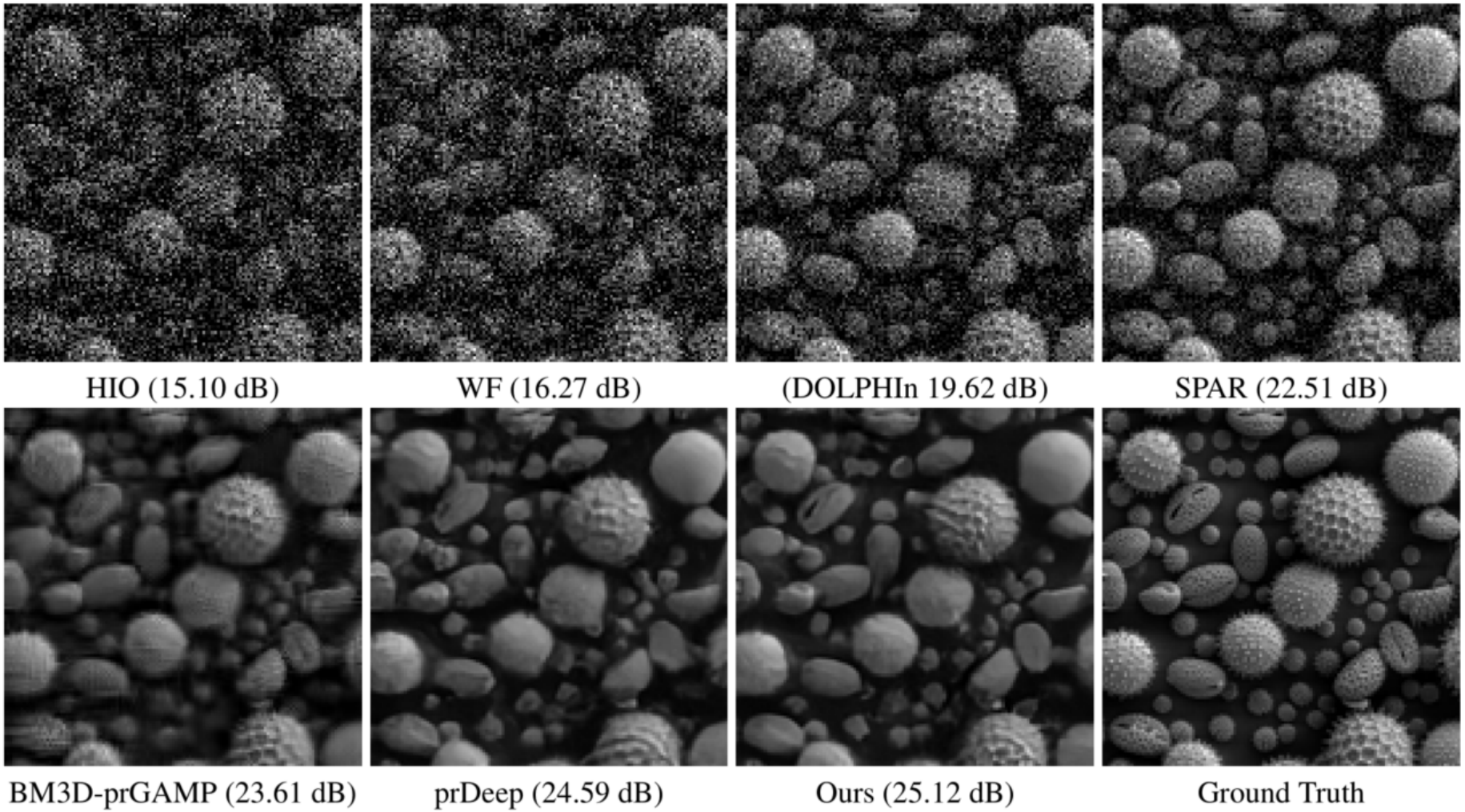
Algorithms	$\alpha = 9$ PSNR	$\alpha = 27$ PSNR	$\alpha = 81$ PSNR
HIO	35.96	25.76	14.82
WF	34.46	24.96	15.76
DOLPHIn	29.93	27.45	19.35
SPAR	35.20	31.82	22.44
BM3D-prGAMP	40.25	32.84	25.43
prDeep	39.70	33.54	26.82
Ours	40.33	33.90	27.23

Quantitative results of different PR algorithms on four CDP measurements and varying amount of Possion noise (large α indicates low sigma-to-noise ratio)

Visual results



Visual results



Conclusion

- We introduce RL into the PnP framework, yielding a novel tuning-free PnP proximal algorithm for a wide range of inverse imaging problems

- The main strength of our proposed method is the policy network, which can customize well-suited parameters for different images

Tuning-free Plug-and-Play Proximal Algorithm for Inverse Imaging Problems

Kaixuan Wei, Angelica Aviles-Rivero, Jingwei Liang,
Ying Fu, Carola-Bibiane Schnlieb, Hua Huang

Code is available at:
https://github.com/Vandermode/TFPnP

Email:
kaixuan_wei@bit.edu.cn