Autotuning Plug-and-Play Algorithms for MRI

Saurav K. Shastri, Rizwan Ahmad, and Philip Schniter

The Ohio State University

This work was supported in part by NIH-R01HL135489.

2020 Asilomar Conference on Signals, Systems, and Computers

Outline

- Background
  - The Linear Regression Problem
  - What is Plug-and-Play?
  - PnP-PDS Algorithm
- Autotuning PnP-PDS Algorithms
  - Must tune step size parameter!
  - Existing work
  - Proposed autotuning algorithm: PDS-ATM2
- Numerical Results
  - Accelerated pMRI image recovery

The Linear Regression Problem

The linear system model of accelerated parallel MRI is:

$$\underbrace{\begin{bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \\ \vdots \\ \boldsymbol{y}_C \end{bmatrix}}_{\boldsymbol{y}} = \underbrace{\begin{bmatrix} \boldsymbol{PFS}_1 \\ \boldsymbol{PFS}_2 \\ \vdots \\ \boldsymbol{PFS}_C \end{bmatrix}}_{\boldsymbol{A}} \boldsymbol{x} + \boldsymbol{w} \qquad \begin{cases} \boldsymbol{P} \in \mathbb{C}^{M\times N}: & \text{sampling matrix } (M \ll N) \\ \boldsymbol{F} \in \mathbb{C}^{N\times N}: & \text{Fourier matrix} \\ \boldsymbol{S}_i \in \mathbb{C}^{N\times N}: & \text{sensitivity map (known)} \\ \boldsymbol{x} \in \mathbb{C}^{N}: & \text{vectorised image} \\ \boldsymbol{w} \in \mathbb{C}^{CM}: & \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}) \end{cases}$$

Goal: Recover $\boldsymbol{x}$ from observations $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}$

Typical methodologies:

- Optimization based algorithms
- Train a deep network to recover $\boldsymbol{x}$ from $\boldsymbol{y}$
- Hybrid: Plug-and-play!

Comparison of various methodologies

- Optimization based algorithms:
  - advantage: simplicity
  - advantage: handles any forward operator
  - disadvantage: poor recovery performance
- Deep networks:
  - advantage: excellent recovery performance
  - disadvantage: requires many fully sampled k-space vectors for training
  - disadvantage: recovery compromised by training/testing mismatch in forward operator
- Plug-and-play:
  - advantage: excellent recovery performance
  - advantage: handles any forward operator
  - advantage: deep-net denoisers can be trained from very few images (via patches)
  - disadvantage: slower prediction speed than some deep networks

Optimization-Based Recovery

- A common approach to recovering image $\boldsymbol{x}$ is through optimization:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \{\ell(\boldsymbol{A}\boldsymbol{x}) + \phi(\boldsymbol{x})\} \text{ with } \begin{cases} \ell(\boldsymbol{A}\boldsymbol{x}): & \text{loss funcion} \\ \phi(\boldsymbol{x}): & \text{regularization} \end{cases}$$

- Assuming both $\ell(\cdot)$ and $\phi(\cdot)$ are convex, people often use:
  - ADMM[1]
  - FISTA[2]
  - primal-dual splitting (PDS[3])

[1] Boyd, Parikh, Chu, Peleato, Eckstein '11, [2] Beck, Teboulle '09, [3] Chambolle, Pock '11.

What is Plug-and-Play?

- ADMM, FISTA, PDS: All alternate a loss-reducing operation with:

$$\text{prox}_{\nu\phi}(\boldsymbol{s}) \triangleq \arg\min_{\boldsymbol{x}} \left\{\phi(\boldsymbol{x}) + \frac{1}{2\nu}\|\boldsymbol{x} - \boldsymbol{s}\|_2^2\right\},$$

  where $\boldsymbol{s}$ and $\nu > 0$ are iteration-dependent parameters

- When $\phi(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$ (Compressed Sensing), prox is soft-thresholding
- The prox can be interpreted as MAP denoising
- "Plug-and-play" (PnP) methods[4,5,6] replace prox with a call to a sophisticated application-specific denoiser, such as BM3D[7] or DnCNN[8]
- Relative to traditional optimization-based approaches, the PnP methods tend to provide much cleaner image recoveries
- In this work, we focus on the PDS variant of PnP[6]

[4] Venkatakrishnan, Bouman, Wohlberg '13, [5] Sun, Wohlberg, Kamilov '18, [6] Ono '17, [7] Dabov, Foi, Katkovnik, Egiazarian '07, [8] Zhang, Zuo, Chen, Meng, Zhang '17.

PnP-PDS Algorithm

- The traditional PnP-PDS algorithm alternates the following two steps:

$$\begin{aligned} \boldsymbol{x}_k &= \boldsymbol{f}(\boldsymbol{x}_{k-1} - \gamma_1 \boldsymbol{A}^{\mathsf{H}} \boldsymbol{v}_{k-1}), \\ \boldsymbol{v}_k &= \operatorname{prox}_{\gamma_2 \ell^*}(\boldsymbol{v}_{k-1} + \gamma_2 \boldsymbol{A}(2\boldsymbol{x}_k - \boldsymbol{x}_{k-1})) \end{aligned}$$

$$\begin{cases} \boldsymbol{f}: & \text{denoiser } \mathbb{C}^N \to \mathbb{C}^N \\ \gamma_1: & \text{tunable positive step size} \\ \gamma_2: & \text{positive step size; } \gamma_2 \leq 1/(\gamma_1 \|\boldsymbol{A}\|_2^2) \\ \ell^*(\cdot): & \text{convex conjugate of loss } \ell(\cdot) \end{cases}$$

- Advantages of PnP-PDS:
  - For most $\ell(\cdot)$, yields first-order algorithm (like FISTA, unlike ADMM)
  - No upper bound on step size γ_1 (like ADMM, unlike FISTA)

  Best of both worlds!

Outline

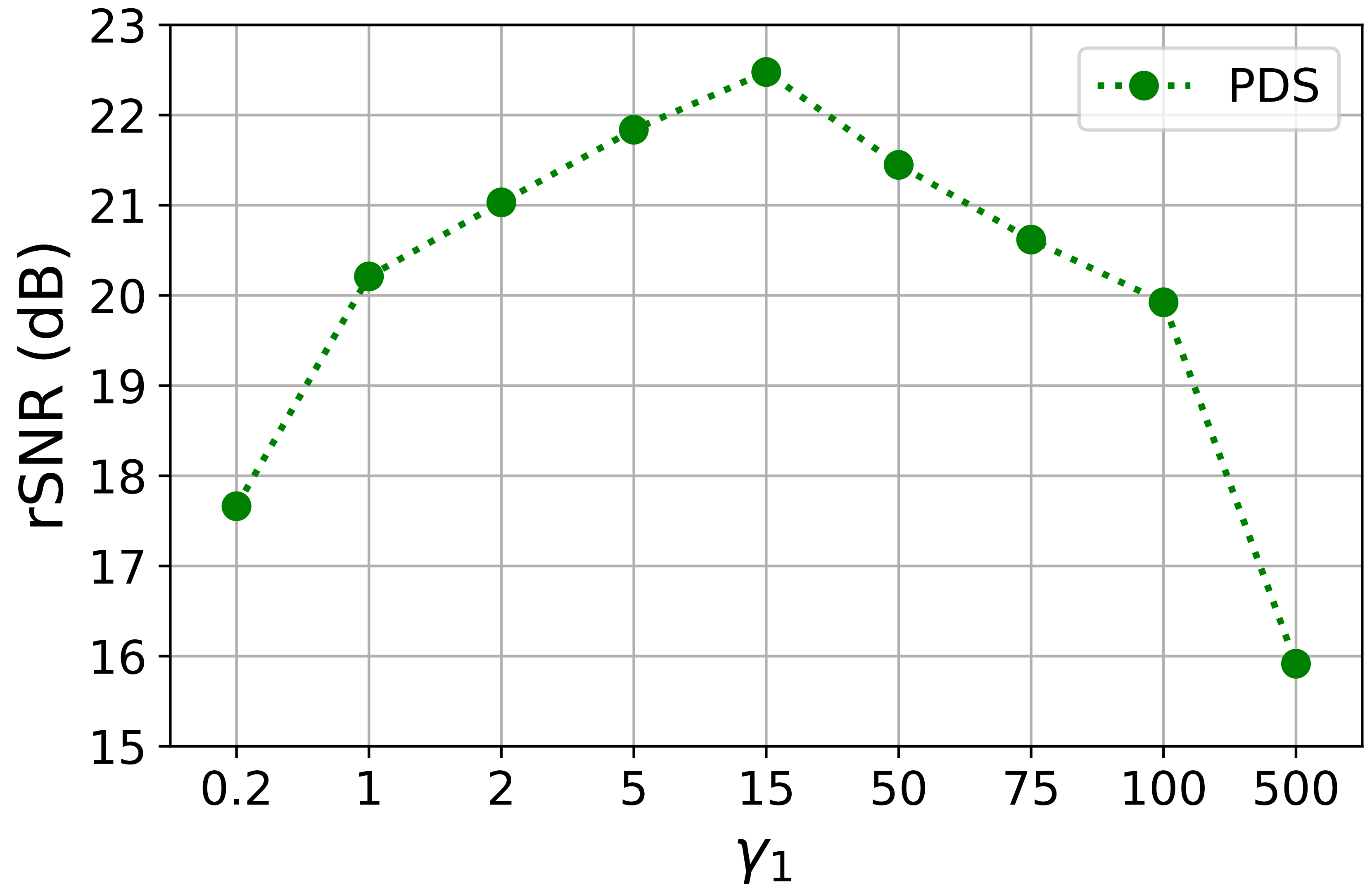
- Background
  - The Linear Regression Problem
  - What is Plug-and-Play?
  - PnP-PDS Algorithm
- Autotuning PnP-PDS Algorithms
  - Must tune step size parameter!
  - Existing work
  - Proposed autotuning algorithm: PDS-ATM2
- Numerical Results
  - Accelerated pMRI image recovery

Must tune γ_1!

- For quadratic data-fidelity loss, i.e., $\ell(\boldsymbol{z}) = \frac{1}{2}\|\boldsymbol{z}-\boldsymbol{y}\|_2^2$, the PnP-PDS fixed point $\widehat{\boldsymbol{x}}$ solves:

$$\widehat{\boldsymbol{x}} = \boldsymbol{f}\left(\widehat{\boldsymbol{x}} - \gamma_1 \boldsymbol{A}^{\mathsf{H}}(\boldsymbol{A}\widehat{\boldsymbol{x}} - \boldsymbol{y})\right)$$

- $\widehat{\boldsymbol{x}}$ is affected by the choice of the stepsize γ_1! (but not γ_2)
- Thus, γ_1 must be tuned to provide good recovery performance
- The goal of our approach is to autotune γ_1



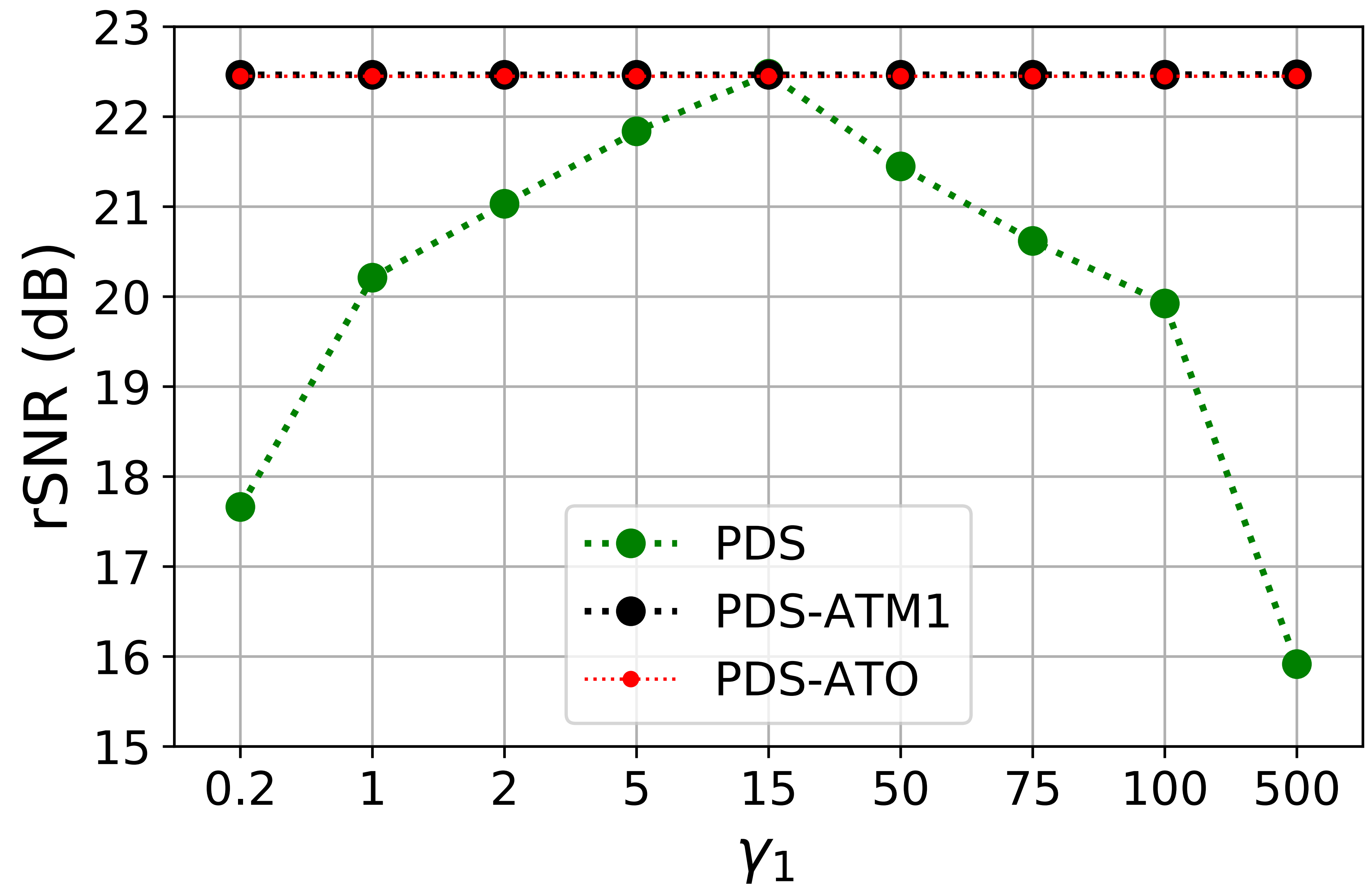
Autotuning PnP-PDS

- It has been proposed[6,9] to tune γ_1 using Morozov's discrepancy principle:

$$\frac{1}{CM}\|\boldsymbol{y} - \boldsymbol{A}\widehat{\boldsymbol{x}}\|_2^2 \approx \beta\sigma^2,$$

where σ^2 is the measurement-noise variance and $\beta \approx 0.95$

- In MRI applications, it is possible to accurately estimate σ^2 from a pre-scan!
- Existing work:
  1. PDS-ATO[6]
  2. PDS-ATM1[9]



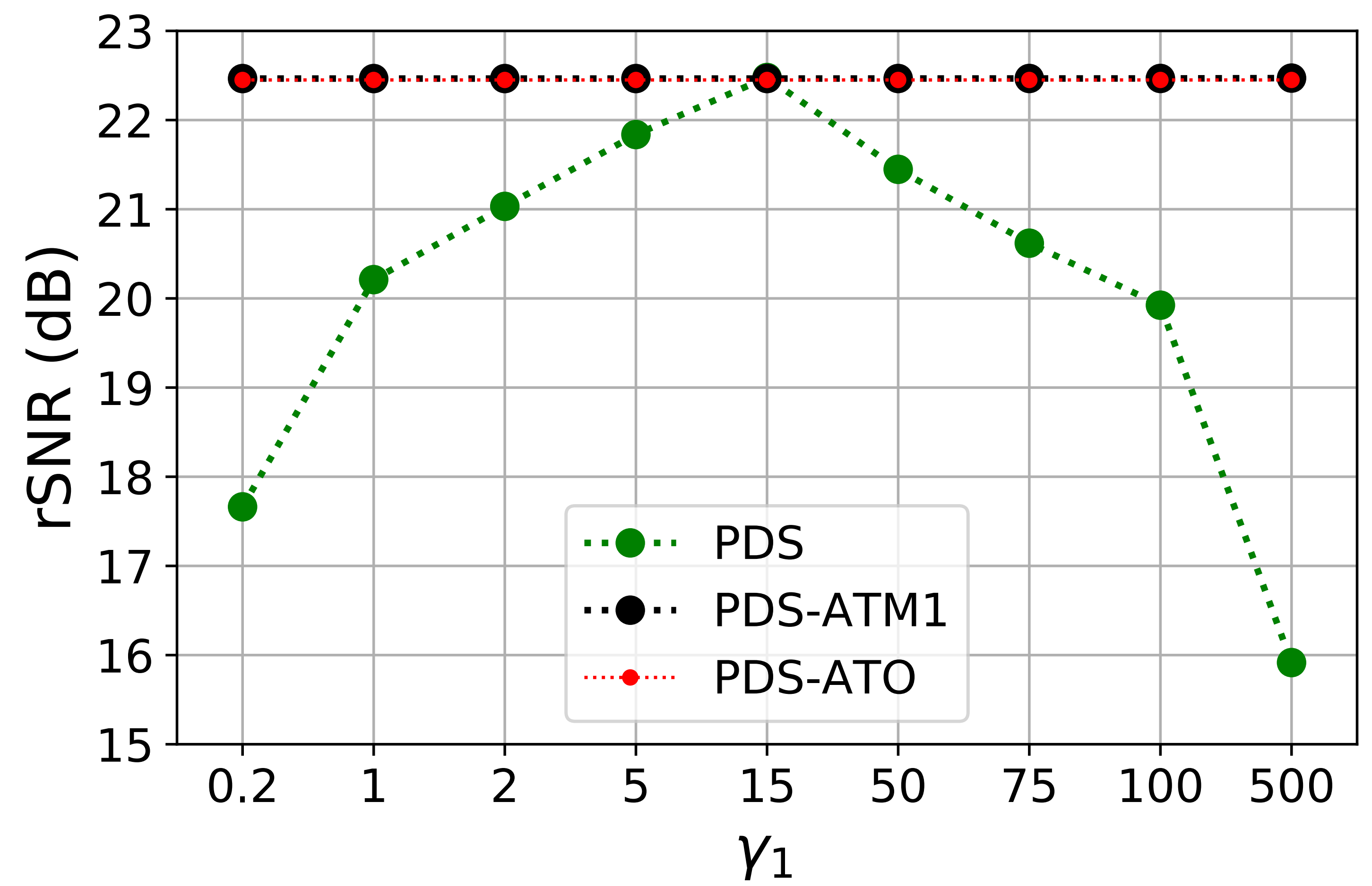
[6] Ono '17, [9] Liu, Jin, Schniter, Ahmad '20.

Autotuning PnP-PDS

- It has been proposed[6,9] to tune γ_1 using Morozov's discrepancy principle:

$$\frac{1}{CM}\|\boldsymbol{y} - \boldsymbol{A}\widehat{\boldsymbol{x}}\|_2^2 \approx \beta\sigma^2,$$

where σ^2 is the measurement-noise variance and $\beta \approx 0.95$



- In MRI applications, it is possible to accurately estimate σ^2 from a pre-scan!
- Existing work:
  1. PDS-ATO[6]
  2. PDS-ATM1[9]

Shunsuke Ono[6] proposed to use PnP-PDS with the indicator loss:

$$\ell(\boldsymbol{z};\boldsymbol{y}) = \begin{cases} 0 & \frac{1}{CM}\|\boldsymbol{y}-\boldsymbol{z}\|^2 \leq \beta\sigma^2 \\ \infty & \text{otherwise} \end{cases}$$

This yields the following algorithm:

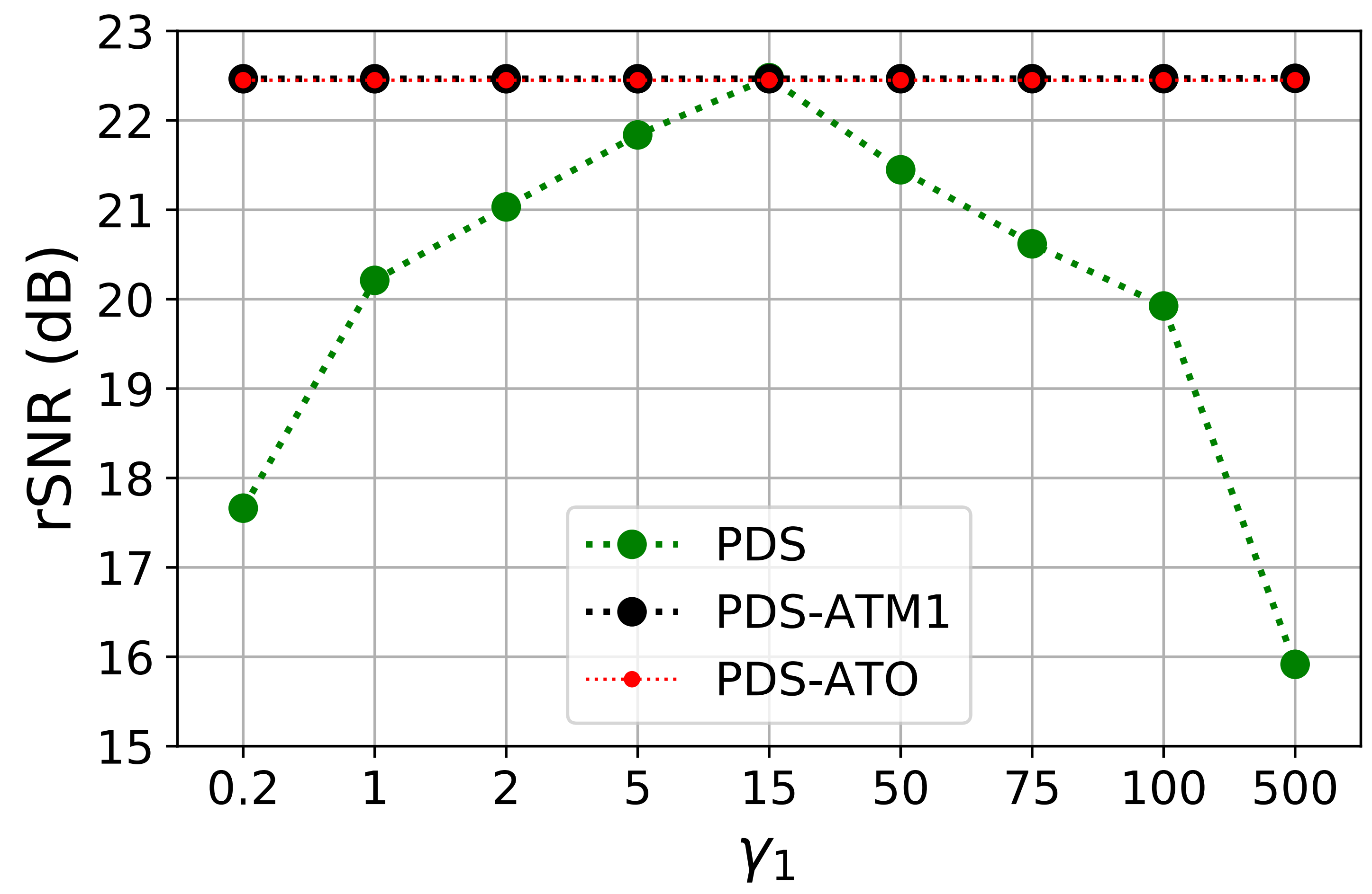
$$\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1} - \gamma_1 \boldsymbol{A}^{\mathsf{H}}\boldsymbol{v}_{k-1}),$$

$$\boldsymbol{v}_k = \max\left\{0, 1 - \frac{\gamma_2\sqrt{\beta CM}\sigma}{\|\boldsymbol{v}_{k-1} + \gamma_2\boldsymbol{A}(2\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) - \gamma_2\boldsymbol{y}\|_2}\right\}(\boldsymbol{v}_{k-1} + \gamma_2\boldsymbol{A}(2\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) - \gamma_2\boldsymbol{y})$$

[6] Ono '17, [9] Liu, Jin, Schniter, Ahmad '20.

graphics courtesy of freepik.com.

Autotuning PnP-PDS



- It has been proposed[6,9] to tune γ_1 using Morozov's discrepancy principle:

$$\frac{1}{CM}\|\boldsymbol{y} - \boldsymbol{A}\widehat{\boldsymbol{x}}\|_2^2 \approx \beta\sigma^2,$$

  where σ^2 is the measurement-noise variance and $\beta \approx 0.95$

- In MRI applications, it is possible to accurately estimate σ^2 from a pre-scan!

- Existing work:
  1. PDS-ATO[6]
  2. PDS-ATM1[9]

Using the quadratic data-fidelity loss, i.e., $\ell(\boldsymbol{z}) = \frac{1}{2}\|\boldsymbol{z} - \boldsymbol{y}\|_2^2$, Liu et al.[9] proposed to adapt γ_1 and γ_2 in PnP-PDS:

$$\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1} - \gamma_{1,k-1}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{v}_{k-1}),$$

$$\boldsymbol{v}_k = \frac{1}{1+\gamma_{2,k-1}}\boldsymbol{v}_{k-1} + \frac{\gamma_{2,k-1}}{1+\gamma_{2,k-1}}(\boldsymbol{A}(2\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) - \boldsymbol{y}),$$

$$\gamma_{1,k} = \gamma_{1,k-1}\left(\frac{\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2}{\beta CM\sigma^2}\right),$$

→ Multiplicative Update

$$\gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$$

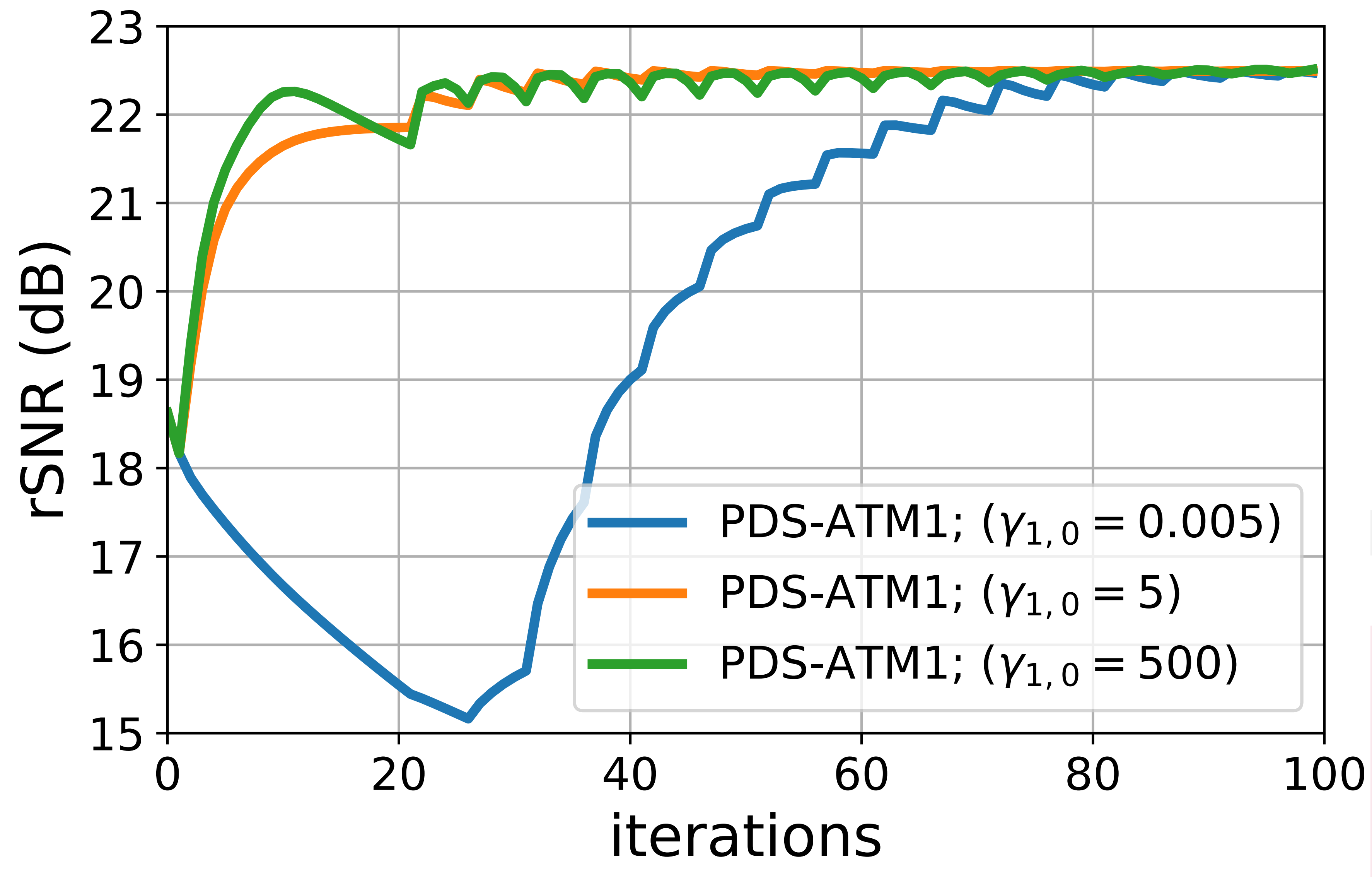
[6] Ono '17, [9] Liu, Jin, Schniter, Ahmad '20.

PDS-ATM1: Existing work

Algorithm 1 The PDS-ATM1 algorithm (Implementation from Liu et al.[9])

Require: $\gamma_1 > 0, \beta \in (0,1], \boldsymbol{x}_0 \in \mathbb{C}^N, \boldsymbol{v}_0 \in \mathbb{C}^N$

```
γ_{1,0} = γ_1,  γ_{2,0} = 1/(γ_1 ||A||_2^2)
for k = 1, 2, 3, ... do
    x_k = f(x_{k-1} − γ_{1,k-1} A^H v_{k-1})
    v_k = 1/(1+γ_{2,k-1}) v_{k-1} + γ_{2,k-1}/(1+γ_{2,k-1}) (A(2x_k − x_{k-1}) − y)
    if k > 19 and mod(k,5) == 0 then
        γ_{1,k} = γ_{1,k-1} ( ||y − A x_k||_2^2 / (β C M σ^2) ),  γ_{2,k} = 1/(γ_{1,k} ||A||_2^2)
    else
        γ_{1,k} = γ_{1,k-1},  γ_{2,k} = γ_{2,k-1}
    end if
end for
```



rSNR (dB) vs. iterations for a test image for various $\gamma_{1,0}$ initialization

- Advantage: γ_1 successfully autotuned
- Disadvantage: convergence speed depends on $\gamma_{1,0}$ initialization
- Disadvantage: Some values of $\gamma_{1,0}$ lead to divergence

[9] Liu, Jin, Schniter, Ahmad '20.

PDS-ATM2: Our refinement

Algorithm 2 The proposed PDS-ATM2 algorithm

Require: $\gamma_1 > 0, \alpha \in (0,1], \beta \in (0,1], \boldsymbol{x}_0 \in \mathbb{C}^N, \boldsymbol{v}_0 \in \mathbb{C}^N$, restart-now = FALSE, restart-allowed = TRUE

$\gamma_{1,0} = \gamma_1,\ \gamma_{2,0} = \frac{1}{\gamma_1 \|\boldsymbol{A}\|_2^2}$

Initialization

for $k = 1, 2, 3, \ldots$ **do**

  $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1} - \gamma_{1,k-1}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{v}_{k-1})$

  $\boldsymbol{v}_k = \frac{1}{1+\gamma_{2,k-1}}\boldsymbol{v}_{k-1} + \frac{\gamma_{2,k-1}}{1+\gamma_{2,k-1}}(\boldsymbol{A}(2\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) - \boldsymbol{y})$

  if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 < \beta C M \sigma^2$ **then**

    restart-allowed = FALSE

  else if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 > 1.1\beta C M \sigma^2$ **then**

    restart-allowed = TRUE

  end if

  if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2 > \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_{k-1}\|_2$ **then**

    restart-now = TRUE

  else

    restart-now = FALSE

  end if

  if restart-now == TRUE **and** restart-allowed == TRUE **then**

    $\gamma_{1,k} = \gamma_{1,0},\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$

  else

    $\gamma_{1,k} = \gamma_{1,k-1}\left(1 + \alpha\left(\frac{\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2}{\beta C M \sigma^2} - 1\right)\right),\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$

  end if

  if $k > 2$ **and** $\gamma_{1,k} == \gamma_{1,k-1} == \gamma_{1,k-2}$ **then**

    $\gamma_{1,0} = 10\gamma_{1,0}$

  end if

end for

PDS-ATM2: Our refinement

Algorithm 2 The proposed PDS-ATM2 algorithm

Require: $\gamma_1 > 0, \alpha \in (0,1], \beta \in (0,1], \boldsymbol{x}_0 \in \mathbb{C}^N, \boldsymbol{v}_0 \in \mathbb{C}^N,$
restart-now = FALSE, restart-allowed = TRUE

$\gamma_{1,0} = \gamma_1,\ \gamma_{2,0} = \frac{1}{\gamma_1 \|\boldsymbol{A}\|_2^2}$

for $k = 1, 2, 3, \ldots$ **do**

$\quad \boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1} - \gamma_{1,k-1}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{v}_{k-1})$

$\quad \boldsymbol{v}_k = \frac{1}{1+\gamma_{2,k-1}}\boldsymbol{v}_{k-1} + \frac{\gamma_{2,k-1}}{1+\gamma_{2,k-1}}(\boldsymbol{A}(2\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) - \boldsymbol{y})$

PNP-PDS main steps

$\quad$ **if** $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 < \beta C M \sigma^2$ **then**

$\quad\quad$ restart-allowed = FALSE

$\quad$ **else if** $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 > 1.1\beta C M \sigma^2$ **then**

$\quad\quad$ restart-allowed = TRUE

$\quad$ **end if**

$\quad$ **if** $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2 > \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_{k-1}\|_2$ **then**

$\quad\quad$ restart-now = TRUE

$\quad$ **else**

$\quad\quad$ restart-now = FALSE

$\quad$ **end if**

$\quad$ **if** restart-now == TRUE **and** restart-allowed == TRUE **then**

$\quad\quad \gamma_{1,k} = \gamma_{1,0},\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$

$\quad$ **else**

$\quad\quad \gamma_{1,k} = \gamma_{1,k-1}\left(1 + \alpha\left(\frac{\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}_k\|_2^2}{\beta C M \sigma^2} - 1\right)\right),\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$

$\quad$ **end if**

$\quad$ **if** $k > 2$ **and** $\gamma_{1,k} == \gamma_{1,k-1} == \gamma_{1,k-2}$ **then**

$\quad\quad \gamma_{1,0} = 10\gamma_{1,0}$

$\quad$ **end if**

end for

PDS-ATM2: Our refinement

Algorithm 2 The proposed PDS-ATM2 algorithm

Require: $\gamma_1 > 0, \alpha \in (0,1], \beta \in (0,1], \boldsymbol{x}_0 \in \mathbb{C}^N, \boldsymbol{v}_0 \in \mathbb{C}^N,$
restart-now = FALSE, restart-allowed = TRUE

- $\gamma_{1,0} = \gamma_1,\ \gamma_{2,0} = \frac{1}{\gamma_1 \|\boldsymbol{A}\|_2^2}$
- **for** $k = 1, 2, 3, \ldots$ **do**
  - $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1} - \gamma_{1,k-1}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{v}_{k-1})$
  - $\boldsymbol{v}_k = \frac{1}{1+\gamma_{2,k-1}}\boldsymbol{v}_{k-1} + \frac{\gamma_{2,k-1}}{1+\gamma_{2,k-1}}(\boldsymbol{A}(2\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) - \boldsymbol{y})$
  - **if** $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 < \beta C M \sigma^2$ **then**
    - restart-allowed = FALSE
  - **else if** $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 > 1.1\beta C M \sigma^2$ **then**
    - restart-allowed = TRUE
  - **end if**
  - **if** $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2 > \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_{k-1}\|_2$ **then**
    - restart-now = TRUE
  - **else**
    - restart-now = FALSE
  - **end if**
  - **if** restart-now == TRUE **and** restart-allowed == TRUE **then**
    - $\gamma_{1,k} = \gamma_{1,0},\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$
  - **else**
    - $\gamma_{1,k} = \gamma_{1,k-1}\left(1 + \alpha\left(\frac{\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}_k\|_2^2}{\beta C M \sigma^2} - 1\right)\right),\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$
  - **end if**
  - **if** $k > 2$ **and** $\gamma_{1,k} == \gamma_{1,k-1} == \gamma_{1,k-2}$ **then**
    - $\gamma_{1,0} = 10\gamma_{1,0}$
  - **end if**
- **end for**

If residual error is below target value, do not allow restart

PDS-ATM2: Our refinement

Algorithm 2 The proposed PDS-ATM2 algorithm

Require: $\gamma_1 > 0, \alpha \in (0,1], \beta \in (0,1], \boldsymbol{x}_0 \in \mathbb{C}^N, \boldsymbol{v}_0 \in \mathbb{C}^N$,
restart-now = FALSE, restart-allowed = TRUE

$\gamma_{1,0} = \gamma_1,\ \gamma_{2,0} = \frac{1}{\gamma_1 \|\boldsymbol{A}\|_2^2}$
for $k = 1, 2, 3, \ldots$ **do**
  $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1} - \gamma_{1,k-1}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{v}_{k-1})$
  $\boldsymbol{v}_k = \frac{1}{1+\gamma_{2,k-1}}\boldsymbol{v}_{k-1} + \frac{\gamma_{2,k-1}}{1+\gamma_{2,k-1}}(\boldsymbol{A}(2\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) - \boldsymbol{y})$
  if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 < \beta C M \sigma^2$ **then**
    restart-allowed = FALSE
  else if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 > 1.1\beta C M \sigma^2$ **then**
    restart-allowed = TRUE
  end if
  if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2 > \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_{k-1}\|_2$ **then**
    restart-now = TRUE
  else
    restart-now = FALSE
  end if
  if restart-now == TRUE **and** restart-allowed == TRUE **then**
    $\gamma_{1,k} = \gamma_{1,0},\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$
  else
    $\gamma_{1,k} = \gamma_{1,k-1}\left(1 + \alpha\left(\frac{\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}_k\|_2^2}{\beta C M \sigma^2} - 1\right)\right),\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$
  end if
  if $k > 2$ **and** $\gamma_{1,k} == \gamma_{1,k-1} == \gamma_{1,k-2}$ **then**
    $\gamma_{1,0} = 10\gamma_{1,0}$
  end if
end for

If the residual error is increasing, restart now (if allowed)

PDS-ATM2: Our refinement

Algorithm 2 The proposed PDS-ATM2 algorithm

Require: $\gamma_1 > 0, \alpha \in (0,1], \beta \in (0,1], \boldsymbol{x}_0 \in \mathbb{C}^N, \boldsymbol{v}_0 \in \mathbb{C}^N$,
restart-now = FALSE, restart-allowed = TRUE

```
γ_{1,0} = γ_1, γ_{2,0} = 1/(γ_1 ||A||_2^2)
for k = 1, 2, 3, ... do
    x_k = f(x_{k-1} − γ_{1,k-1} A^H v_{k-1})
    v_k = 1/(1+γ_{2,k-1}) v_{k-1} + γ_{2,k-1}/(1+γ_{2,k-1}) (A(2x_k − x_{k-1}) − y)
    if ||y − A x_k||_2^2 < βCMσ^2 then
        restart-allowed = FALSE
    else if ||y − A x_k||_2^2 > 1.1βCMσ^2 then
        restart-allowed = TRUE
    end if
    if ||y − A x_k||_2 > ||y − A x_{k-1}||_2 then
        restart-now = TRUE
    else
        restart-now = FALSE
    end if
    if restart-now == TRUE and restart-allowed == TRUE then
        γ_{1,k} = γ_{1,0}, γ_{2,k} = 1/(γ_{1,k} ||A||_2^2)
    else
        γ_{1,k} = γ_{1,k-1} (1 + α(||y − A x_k||_2^2 / (βCMσ^2) − 1)), γ_{2,k} = 1/(γ_{1,k} ||A||_2^2)
    end if
    if k > 2 and γ_{1,k} == γ_{1,k-1} == γ_{1,k-2} then
        γ_{1,0} = 10γ_{1,0}
    end if
end for
```

Either restart step sizes with initial values or multiplicatively update them

PDS-ATM2: Our refinement

Algorithm 2 The proposed PDS-ATM2 algorithm

Require: $\gamma_1 > 0, \alpha \in (0,1], \beta \in (0,1], \boldsymbol{x}_0 \in \mathbb{C}^N, \boldsymbol{v}_0 \in \mathbb{C}^N,$
restart-now = FALSE, restart-allowed = TRUE

$\gamma_{1,0} = \gamma_1,\ \gamma_{2,0} = \frac{1}{\gamma_1 \|\boldsymbol{A}\|_2^2}$
for $k = 1, 2, 3, \ldots$ **do**
  $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1} - \gamma_{1,k-1}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{v}_{k-1})$
  $\boldsymbol{v}_k = \frac{1}{1+\gamma_{2,k-1}}\boldsymbol{v}_{k-1} + \frac{\gamma_{2,k-1}}{1+\gamma_{2,k-1}}(\boldsymbol{A}(2\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) - \boldsymbol{y})$
  if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 < \beta C M \sigma^2$ **then**
    restart-allowed = FALSE
  else if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 > 1.1\beta C M \sigma^2$ **then**
    restart-allowed = TRUE
  end if
  if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2 > \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_{k-1}\|_2$ **then**
    restart-now = TRUE
  else
    restart-now = FALSE
  end if
  if restart-now == TRUE **and** restart-allowed == TRUE **then**
    $\gamma_{1,k} = \gamma_{1,0},\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$
  else
    $\gamma_{1,k} = \gamma_{1,k-1}\left(1 + \alpha\left(\frac{\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}_k\|_2^2}{\beta C M \sigma^2} - 1\right)\right),\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$
  end if
  if $k > 2$ **and** $\gamma_{1,k} == \gamma_{1,k-1} == \gamma_{1,k-2}$ **then**
    $\gamma_{1,0} = 10\gamma_{1,0}$
  end if
end for

If third restart, increase initial $\gamma_{1,0}$

PDS-ATM2: Our refinement

Algorithm 2 The proposed PDS-ATM2 algorithm

Require: $\gamma_1 > 0, \alpha \in (0,1], \beta \in (0,1], \boldsymbol{x}_0 \in \mathbb{C}^N, \boldsymbol{v}_0 \in \mathbb{C}^N$, restart-now = FALSE, restart-allowed = TRUE

$\gamma_{1,0} = \gamma_1,\ \gamma_{2,0} = \frac{1}{\gamma_1 \|\boldsymbol{A}\|_2^2}$

for $k = 1, 2, 3, \dots$ **do**

  $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1} - \gamma_{1,k-1}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{v}_{k-1})$

  $\boldsymbol{v}_k = \frac{1}{1+\gamma_{2,k-1}}\boldsymbol{v}_{k-1} + \frac{\gamma_{2,k-1}}{1+\gamma_{2,k-1}}(\boldsymbol{A}(2\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) - \boldsymbol{y})$

  if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 < \beta C M \sigma^2$ **then**

    restart-allowed = FALSE

  else if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 > 1.1\beta C M \sigma^2$ **then**

    restart-allowed = TRUE

  end if

  if $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_k\|_2 > \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_{k-1}\|_2$ **then**

    restart-now = TRUE

  else

    restart-now = FALSE

  end if

  if restart-now == TRUE **and** restart-allowed == TRUE **then**

    $\gamma_{1,k} = \gamma_{1,0},\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$

  else

    $\gamma_{1,k} = \gamma_{1,k-1}\left(1 + \alpha\left(\frac{\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}_k\|_2^2}{\beta C M \sigma^2} - 1\right)\right),\ \gamma_{2,k} = \frac{1}{\gamma_{1,k}\|\boldsymbol{A}\|_2^2}$

  end if

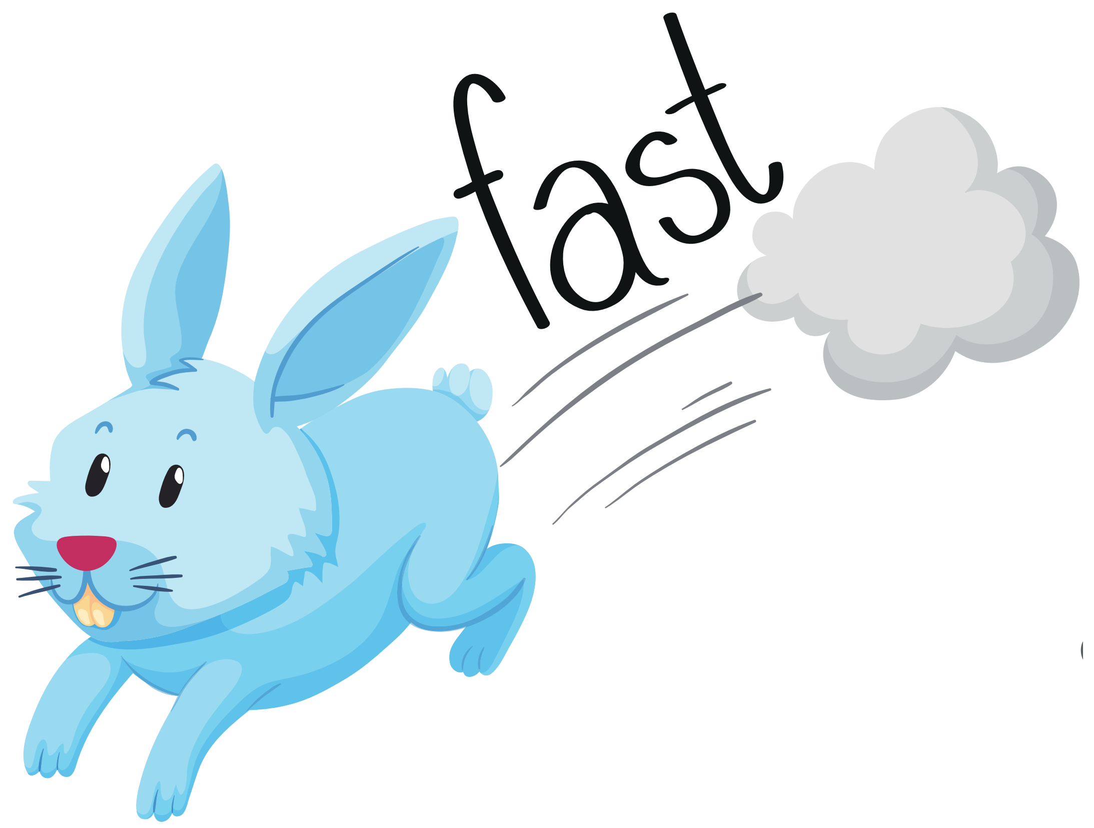
  if $k > 2$ **and** $\gamma_{1,k} == \gamma_{1,k-1} == \gamma_{1,k-2}$ **then**

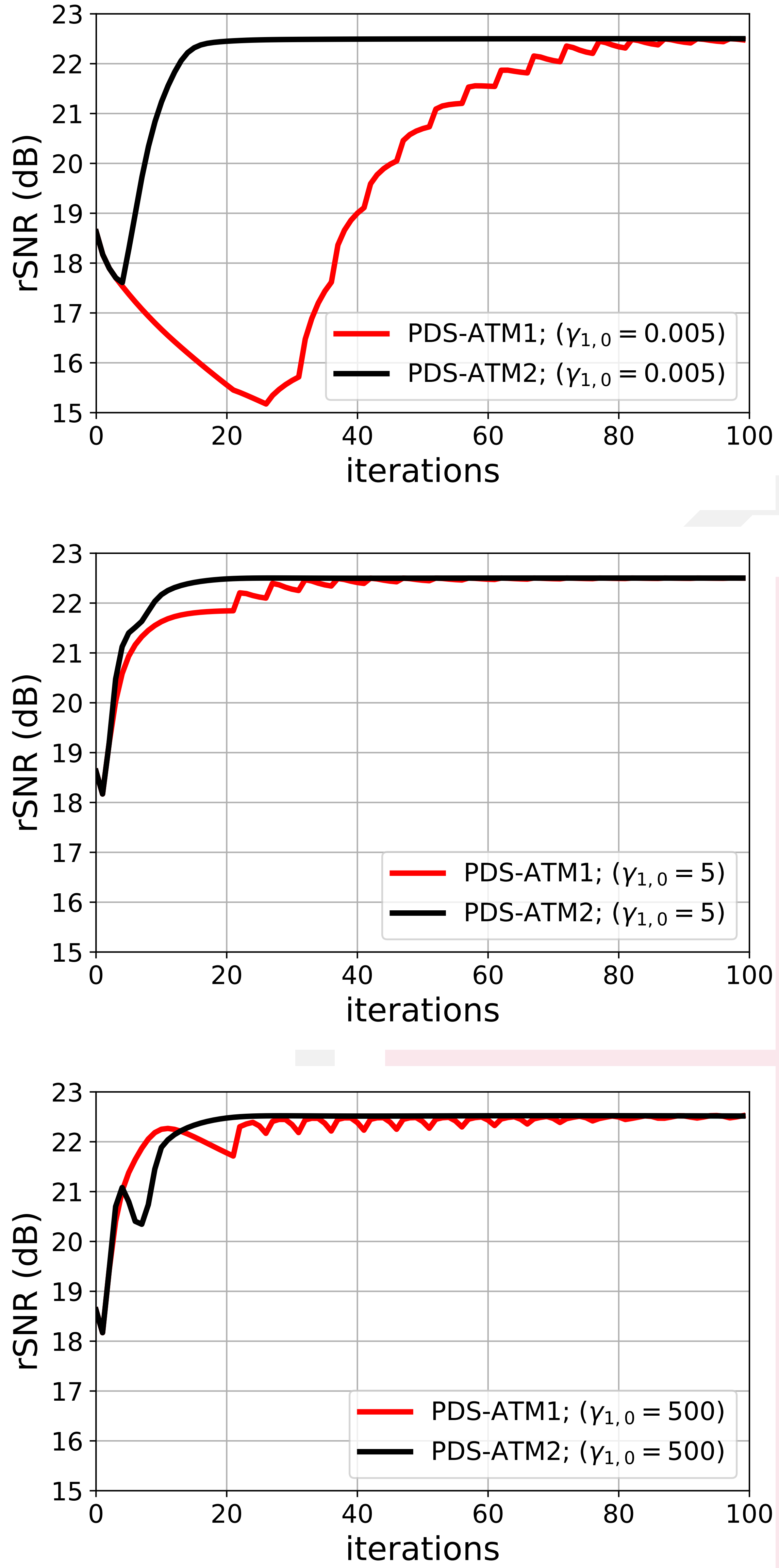
    $\gamma_{1,0} = 10\gamma_{1,0}$

  end if

end for

FAST & ROBUST





graphics courtesy of freepik.com.

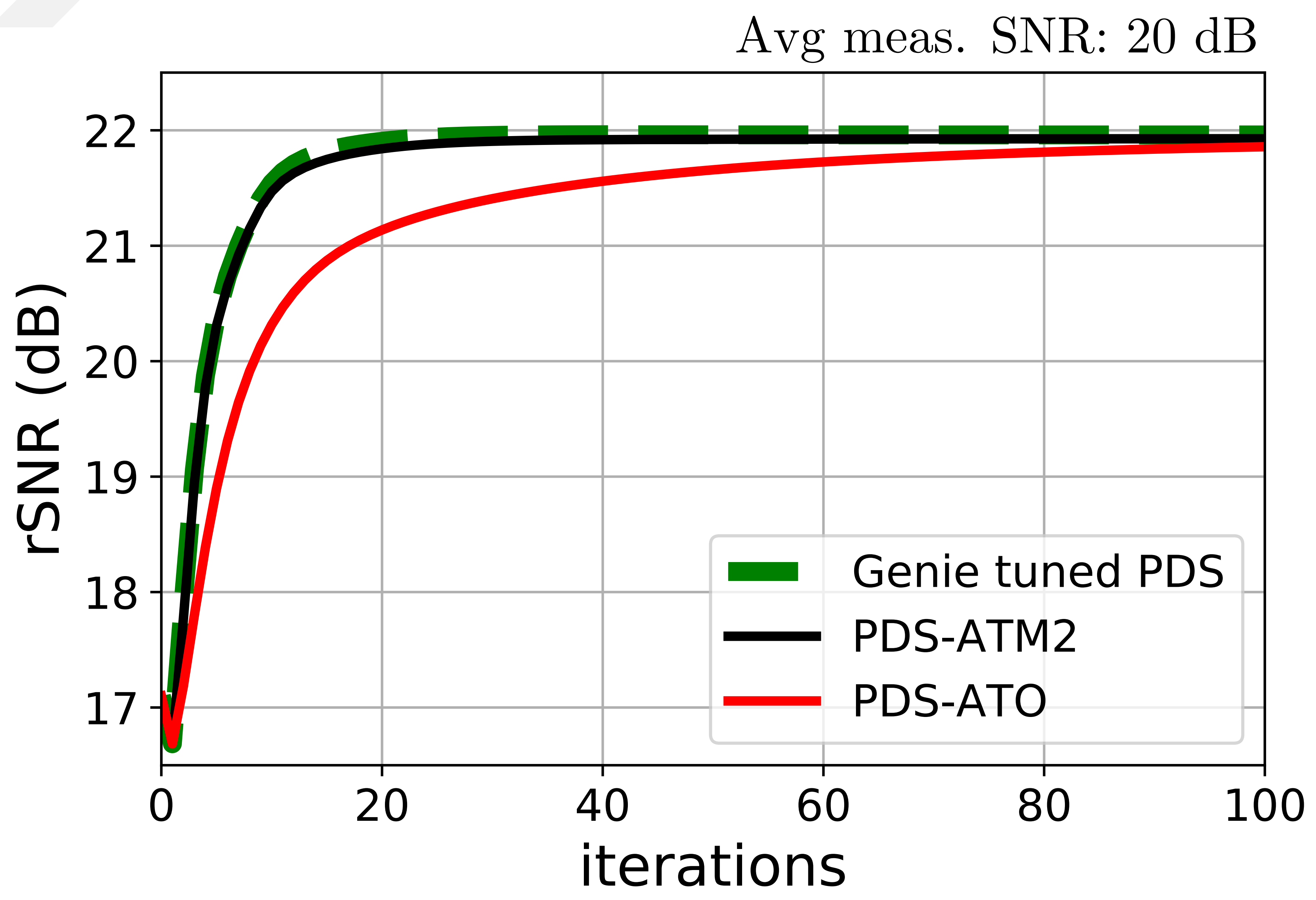
Outline

- Background
  - The Linear Regression Problem
  - What is Plug-and-Play?
  - PnP-PDS Algorithm
- Autotuning PnP-PDS Algorithms
  - Must tune step size parameter!
  - Existing work
  - Proposed autotuning algorithm: PDS-ATM2
- Numerical Results
  - Accelerated pMRI image recovery

Experiment: pMRI image recovery (4 coils; acceleration R = 4)

Setup:

- White Gaussian noise of variance σ^2 was added to attain SNR of 15 or 20 dB
- An MRI-specific deep-convolutional-neural-net denoiser was used[10]
- fastMRI[11] knee images



rSNR (dB) averaged over 10 images vs. iterations

- PDS-ATM2 converges to genie tuned $\widehat{\boldsymbol{x}}$ significantly faster than PDS-ATO!

[10] Ahmad et al. '20., [11] Zbontar et al. '19.

Experiment: pMRI image recovery (4 coils; acceleration R = 4)

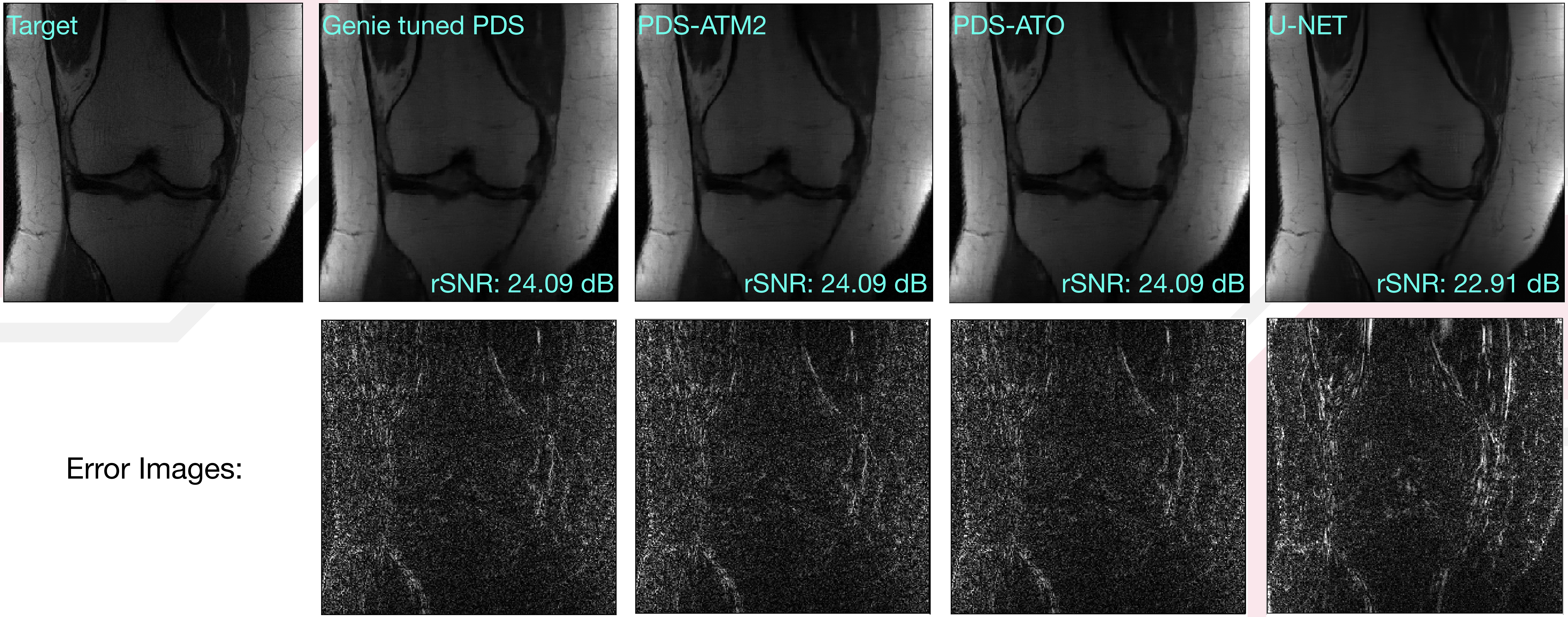
Average recovery SNR (rSNR) and SSIM for 10 images after 2000 iterations

	Avg meas. SNR: 15 dB		Avg meas. SNR: 20 dB	
	rSNR(dB)	SSIM	rSNR(dB)	SSIM
Genie tuned PDS	21.02	0.804	21.96	0.839
PDS-ATM2	21.00	0.805	21.93	0.833
PDS-ATO	21.00	0.805	21.93	0.833
U-Net[11]	20.77	0.838	21.34	0.841

- PDS-ATM2 asymptotically identical to PDS-ATO
- Autotuned PDS very close to genie tuned PDS
- PnP methods very close to UNet

[11] Zbontar et al. '19.

Experiment: pMRI image recovery (4 coils; acceleration R = 4)



Conclusions

- Considered PnP-PDS algorithm for MRI image recovery
- Fixed point of PnP-PDS depends on step size parameter γ_1
- Can autotune γ_1 using Morozov's discrepancy principle for MRI applications!
- Existing autotuning techniques are slow and/or not robust
- We proposed a fast and robust autotuning PnP-PDS for MRI image recovery