Preconditioned Plug-and-Play ADMM with Locally Adjustable Denoiser for Image Restoration

Mikael Le Pendu and Christine Guillemot

Outline

01. Introduction
02. Preconditioned Plug-and-Play ADMM
03. Training of a Locally Adjustable Denoiser
04. Applications and Results
05. Conclusion

01

Introduction

Plug-and-Play priors with ADMM

Inverse problems

- E.g. Linear degradation model with additive Gaussian noise:

$$b = Ax + \sigma \cdot n$$

- b: degraded image
- A: degradation matrix (e.g. subsampling)
- x: (unknown) ground truth image
- $\sigma \cdot n$: additive Gaussian noise (s.t.d. σ)

$$x^* = \underset{x}{\operatorname{argmin}} \frac{1}{2}\|Ax - b\|_2^2 + \sigma^2 \cdot R(x)$$

Data fidelity term

Regularization term

- Represents prior knowledge on images
- Penalizes unlikely solutions, i.e. "unnatural" images

Plug-and-Play priors with ADMM

Inverse problems

- E.g. Linear degradation model with additive Gaussian noise:

$$x^* = \underset{x}{\operatorname{argmin}} \frac{1}{2} \|Ax - b\|_2^2 + \sigma^2 \cdot R(x)$$

- **Question 1: How to define R?**
- **Question 2: How to solve the minimization?**

- In the plug-and-play approach:
  - We answer question 2 first
  - No need to define R explicitly!

Plug-and-Play priors with ADMM

Let's answer Question 2 first: how to solve the minimization?

- Split the two terms using two variables with equality constraint

$$x^* = \underset{x}{\operatorname{argmin}} \frac{1}{2}\|Ax - b\|_2^2 + \sigma^2 \cdot R(x)$$

$$= \underset{\substack{x,y \\ x=y}}{\operatorname{argmin}} \frac{1}{2}\|Ax - b\|_2^2 + \sigma^2 \cdot R(y)$$

  - Equivalent mathematically
  - But now we can decouple the two terms and solve alternately for x and y

Plug-and-Play priors with ADMM

Augmented direction method of multipliers (ADMM)

1. x-subproblem: $x^{k+1} = \underset{x}{\operatorname{argmin}} \frac{1}{2}\|Ax - b\|_2^2 + \frac{\rho}{2} \cdot \left\| x - \left(y^k - \frac{l^k}{\rho} \right) \right\|_2^2$ ➔ has a closed-form solution

   only depends on data term

2. y-subproblem: $y^{k+1} = \underset{y}{\operatorname{argmin}} \frac{1}{2}\|y - u\|_2^2 + \frac{\sigma^2}{\rho} R(y), \quad \text{with u} = \left(x^{k+1} + \frac{l^k}{\rho} \right)$

   only depends on regularization term

3. Dual update: $l^{k+1} = l^k + \rho(x^{x+1} - y^{k+1})$

l : dual variable (helps satisfy the constraint x=y)
ρ: penalty parameter (also ensures the constraint x=y)

Plug-and-Play priors with ADMM

Augmented direction method of multipliers (ADMM)

- y-subproblem (i.e. regularization sub-problem)
  - Similar form as original inverse problem but without degradation matrix (i.e. $A = I$)

No degradation matrix

$$y^{k+1} = \underset{y}{\operatorname{argmin}} \frac{1}{2}\|y - u\|_2^2 + \frac{\sigma^2}{\rho} R(y)$$

➔ Equivalent to Gaussian denoising problem with the prior defined by R and for noise variance $\frac{\sigma^2}{\rho}$

➔ No need to define R explicitly: use instead a Gaussian denoiser to find y^{k+1} directly

e.g. « Plug » in the algorithm a high performance neural net trained for denoising

Limitations of the Plug-and-Play approach

- **No convergence guarantee for an arbitrary denoiser (not convex optimization)**
- **Can be overcome by:**
  - Training a denoiser with desirable properties for better convergence

    ➔ *May reduce accuracy of the underlying image prior*
  - "Unrolling" the algorithm's iterations for end-to-end training of the denoiser

    ➔ *Denoiser trained (or fine-tuned) to optimize the performance of the algorithm for each task*

    ➔ *But: less universal approach (task-specific training).*
  - **Proposed approach:** Apply preconditioning to the algorithm

    ➔ *Adapt the preconditioner to each task for improved convergence and quality*

    ➔ *Preserve the genericity of the plug-and-play approach*

02

Preconditioned Plug-and-Play ADMM

Preconditioned ADMM formulation

Re-writting the original problem

- **Change of variable:** $x = P\tilde{x}$

$$\tilde{x}^* = \underset{\tilde{x}}{\operatorname{argmin}} \frac{1}{2}\|AP\tilde{x} - b\|_2^2 + \sigma^2 \cdot R(P\tilde{x})$$

$$x^* = P\tilde{x}^*$$

- Choose P such that AP is better conditioned than A
  - e.g. Ideally $P = A^{-1}$ ➔ $AP = I$ (but A is not invertible in general)

Preconditioned ADMM formulation

Re-writting the ADMM

1. x-subproblem: $\tilde{x}^{k+1} = \underset{\tilde{x}}{\operatorname{argmin}} \frac{1}{2}\|AP\tilde{x} - b\|_2^2 + \frac{\rho}{2} \cdot \left\| \tilde{x} - \left(y^k - \frac{l^k}{\rho} \right) \right\|_2^2$

2. y-subproblem: $\tilde{y}^{k+1} = \underset{\tilde{y}}{\operatorname{argmin}} \frac{1}{2}\|\tilde{y} - \tilde{u}\|_2^2 + \frac{\sigma^2}{\rho} R(P\tilde{y}),$ with $\tilde{u} = \left(\tilde{x}^{k+1} + \frac{l^k}{\rho} \right)$

3. Dual update $l^{k+1} = l^k + \rho(x^{x+1} - y^{k+1})$

Preconditioned ADMM formulation

Breaking the assumption of independent and identically distributed (i.i.d.) noise

Denoising for i.i.d. Gaussian noise of s.t.d. σ (i.e. variance σ^2)

$$D_R(u_{noisy}, \sigma) = \underset{y}{\operatorname{argmin}} \frac{1}{2} \|y - u_{noisy}\|_2^2 + \sigma^2 \cdot R(y)$$

Denoising for non-i.i.d. Gaussian noise of covariance matrix Σ

$$\widetilde{D}_R(u_{noisy}, \Sigma^{\frac{1}{2}}) = \underset{x}{\operatorname{argmin}} \frac{1}{2} \left\|\Sigma^{-\frac{1}{2}}(x - u_{noisy})\right\|_2^2 + R(x)$$

Preconditioned ADMM formulation

Breaking the assumption of independent and identically distributed (i.i.d.) noise

Denoising for i.i.d. Gaussian noise of s.t.d. σ (i.e. variance σ^2)

$$D_R(u_{noisy}, \sigma) = \underset{y}{\operatorname{argmin}} \frac{1}{2} \|y - u_{noisy}\|_2^2 + \sigma^2 \cdot R(y)$$

Denoising for non-i.i.d. Gaussian noise of covariance matrix Σ

$$\widetilde{D}_R(u_{noisy}, \Sigma^{\frac{1}{2}}) = \underset{y}{\operatorname{argmin}} \frac{1}{2} \left\|\Sigma^{-\frac{1}{2}}(y - u_{noisy})\right\|_2^2 + R(y)$$

$$\tilde{y}^{k+1} = \underset{\tilde{y}}{\operatorname{argmin}} \frac{1}{2} \|\tilde{y} - \tilde{u}\|_2^2 + \frac{\sigma^2}{\rho} R(P\tilde{y}) = P^{-1} \widetilde{D}_R\left(P\tilde{u}, \frac{\sigma}{\sqrt{\rho}} P\right)$$

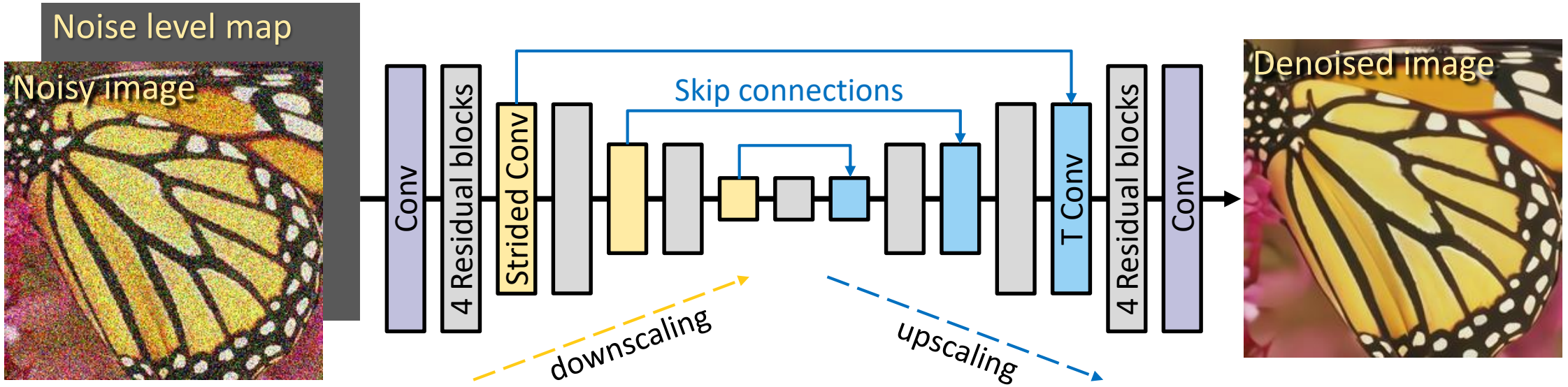
➔ Preconditioning matrix proportional to the square root of covariance matrix

03

Training of a Locally Adjustable Denoiser

State-of-the-art DRUNet Denoiser

DRUNet (combine ResNet and U-Net ideas) with input noise level map [1]

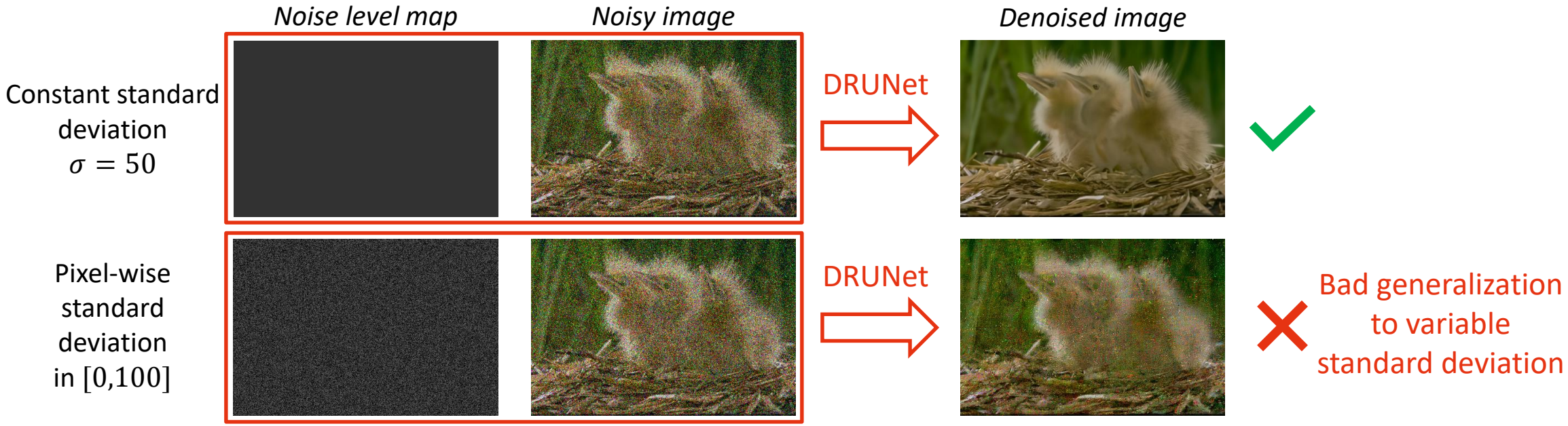


[1] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, **"Plug-and-play image restoration with deep denoiser prior,"** *PAMI*, 2021.

State-of-the-art DRUNet Denoiser

Problem with original DRUNet denoiser

- Trained only for constant noise level maps



Extending for Locally Adjustable Denoising

Noise level map generation for training

- At each pixel i, we generate a random standard deviation value S_i, using:

$$S_i = \alpha \cdot (X_i \cdot (1 - W) + O \cdot W)$$

α: maximum standard deviation

X_i, O, W : random variables with distribution $\mathcal{U}(0,1)$

Extending for Locally Adjustable Denoising

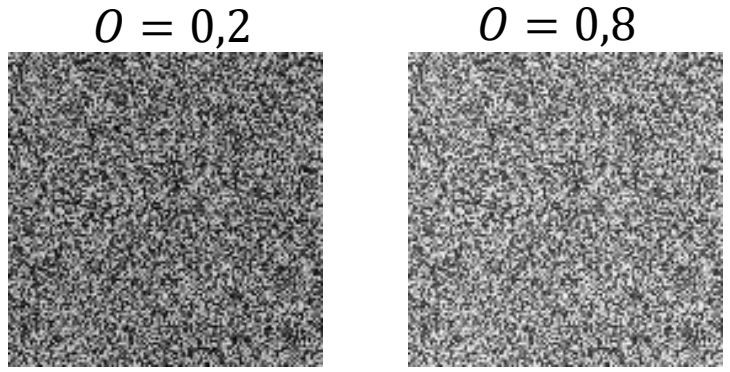
Noise level map generation for training

- At each pixel i, we generate a random standard deviation value S_i, using:

$$S_i = \alpha \cdot (X_i \cdot (1 - W) + O \cdot W)$$

α: maximum standard deviation

X_i, O, W : random variables with distribution $\mathcal{U}(0,1)$



Extending for Locally Adjustable Denoising

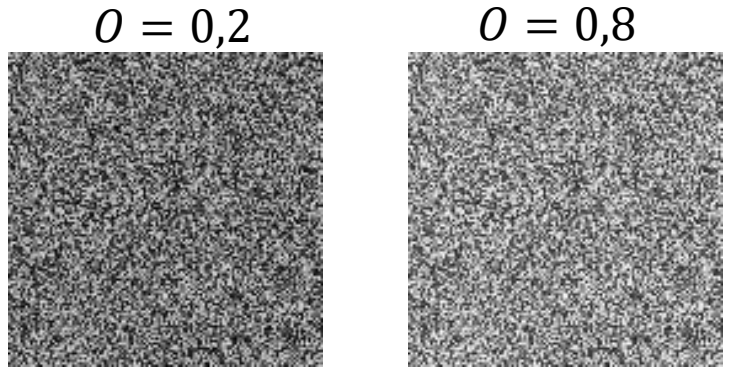
Noise level map generation for training

- At each pixel i, we generate a random standard deviation value S_i, using:

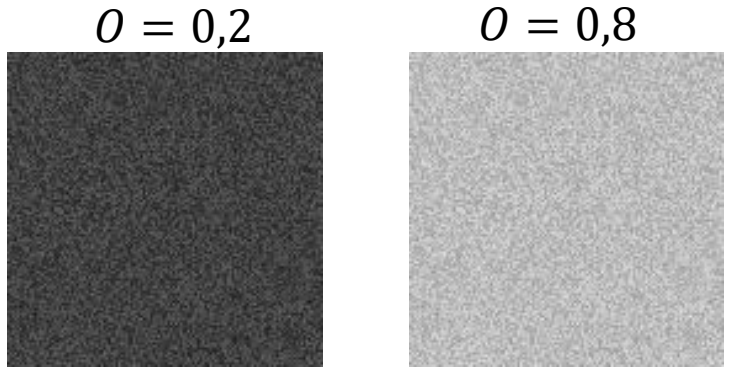
$$S_i = \alpha \cdot (X_i \cdot (1 - W) + O \cdot W)$$

α: maximum standard deviation

X_i, O, W : random variables with distribution $\mathcal{U}(0,1)$



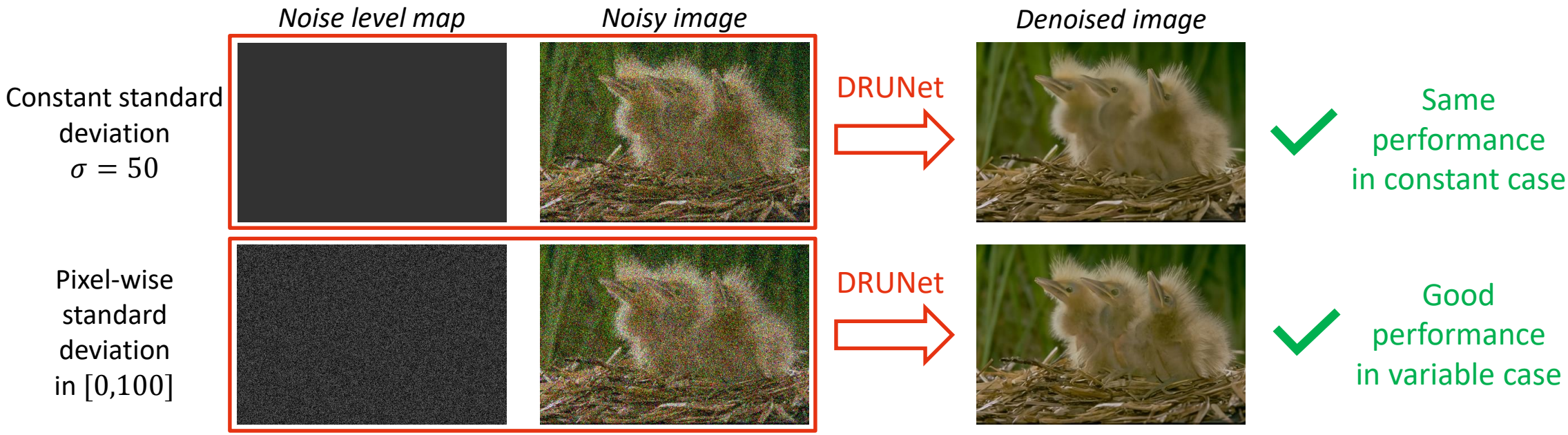
low W $(W = 0{,}2)$ ➔ strong variations



high W $(W = 0{,}8)$ ➔ small variations

Extending for Locally Adjustable Denoising

Result of Locally Adjustable DRUNet denoiser



04

Applications and Results

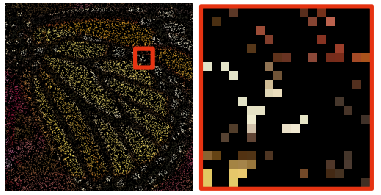
Treated applications

Sampling problems

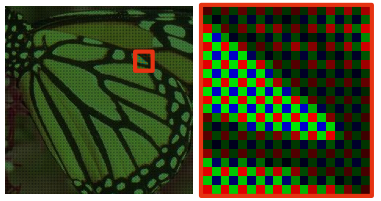
x
(ground truth)

Degradation

➡ Interpolation



➡ Completion



➡ Demosaicing

➔Error variance of the estimates is expected to depend on the pixel position (e.g. no error for sampled pixels)

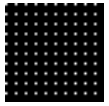
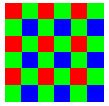
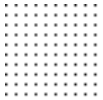
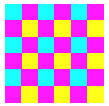
Poisson denoising

➔ Variance of Poisson noise equal to the pixel's value in the ground truth

➡ Poisson denoising

Sampling problems

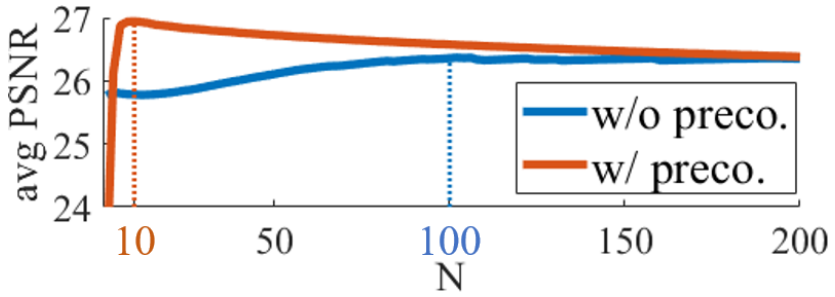
Choice of preconditioning matrix

	interpolation	completion	demosaicing	Diagonal matrix
Sampling mask m				➔ $A_{ii} = m(i)$
$p = \frac{1+\epsilon}{m+\epsilon}$				➔ $P_{ii} = p(i)$

Sampling problems

Interpolation Results

- 4x upsampling



- Without preconditioning the PnP-ADMM fails to recover all the details
- Better and faster recovery with preconditioning

Ground truth

bicubic

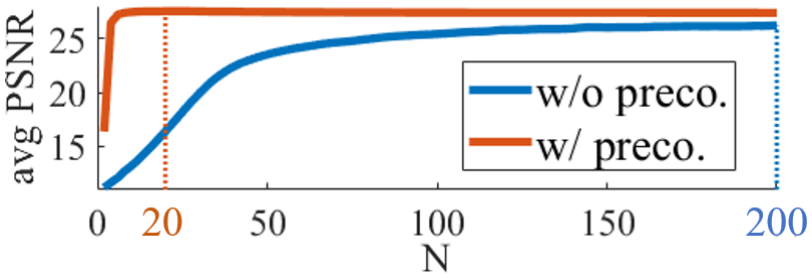
PnP-ADMM w/o preco.
N=100 iterations

PnP-ADMM with preco.
N=10 iterations

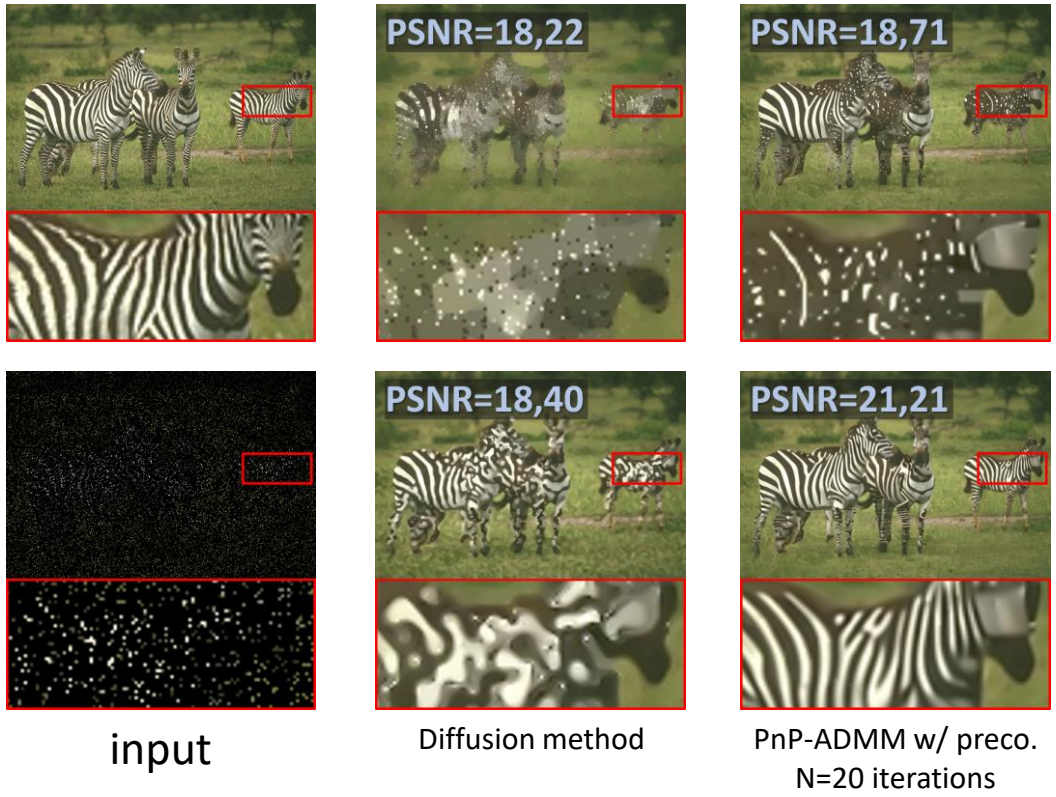
Sampling problems

Completion Results

- 10% known pixels (randomly sampled)



- Without preconditioning the PnP-ADMM fails to recover all the details
- Better and faster recovery with preconditioning

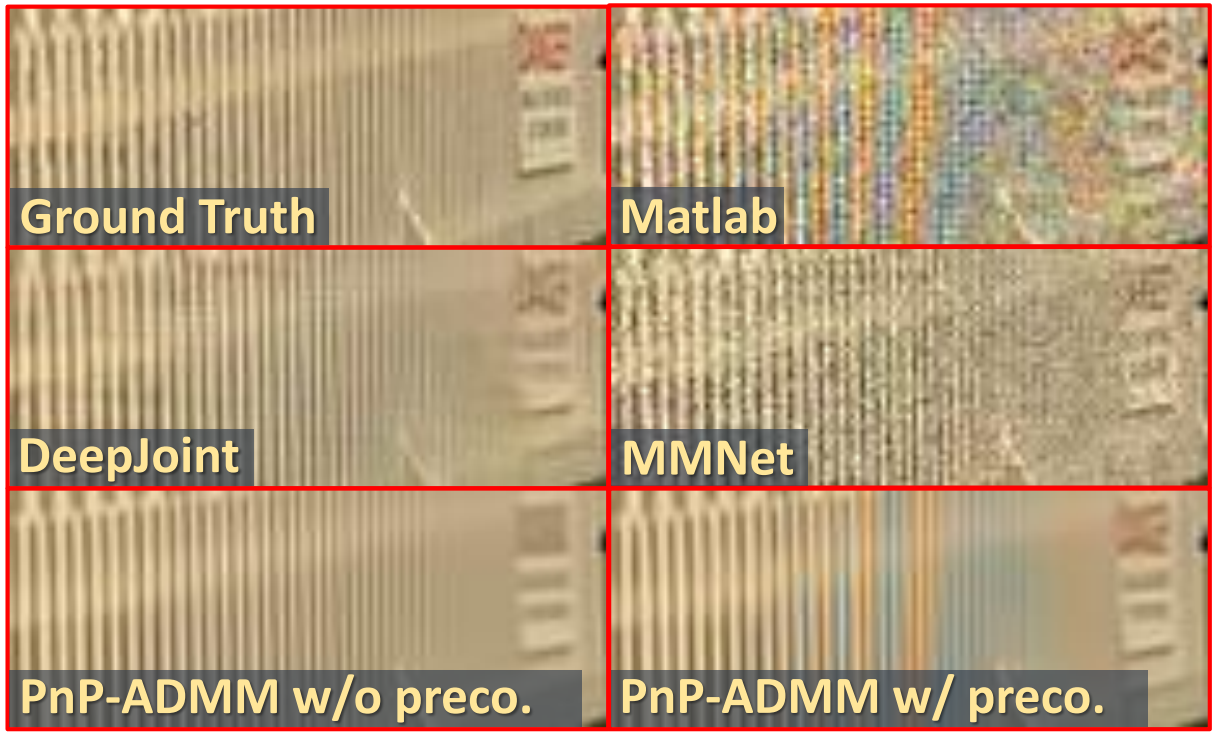


Sampling problems

Joint Demosaicing and Denoising

- Plug-And-Play ADMM better than reference methods (DeepJoint, MMNet)
- More details preserved with preconditioning
- But color artifacts remain in challenging areas

➔ Locally Adjustable Denoiser trained from random maps does not perform well with demosaicing pattern

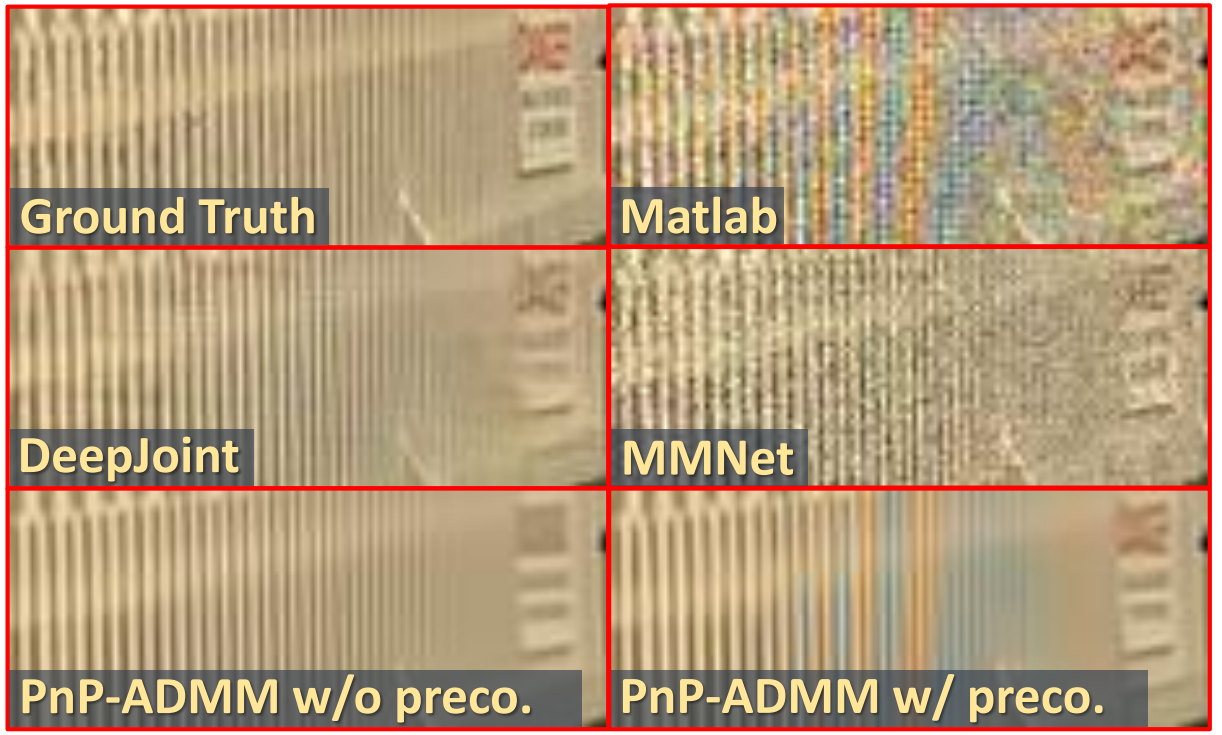


Sampling problems

Joint Demosaicing and Denoising

- Plug-And-Play ADMM better than reference methods (DeepJoint, MMNet)
- More details preserved with preconditioning
- But color artifacts remain in challenging areas

➔ Can be corrected by training a Locally Adjustable Denoiser with the preconditioning pattern used in demosaicing instead of random noise level maps.



Poisson Denoising

Different data fidelity term

Data term for Gaussian noise: $\|x - b\|_2^2$ ➡ Data term for Poisson noise: $-b^\top \ln(x) + \mathbb{1}^\top x$

- Quadratic data fidelity term no longer suitable
- x-subproblem (i.e. related to data term) still has a closed-form solution

Choice of preconditioning matrix

- P proportional to square root of noise covariance matrix Σ in the denoising step
- For an image corrupted with Poisson noise : pixel-wise noise variance = ground truth pixel value

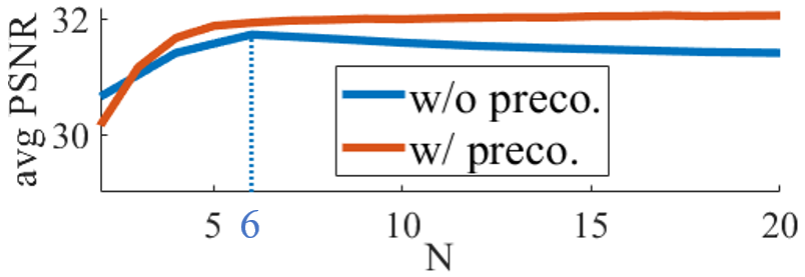
➔ Take $P_{i,i} = \sqrt{b_i}$ *(i.e. square root of the noisy image)*

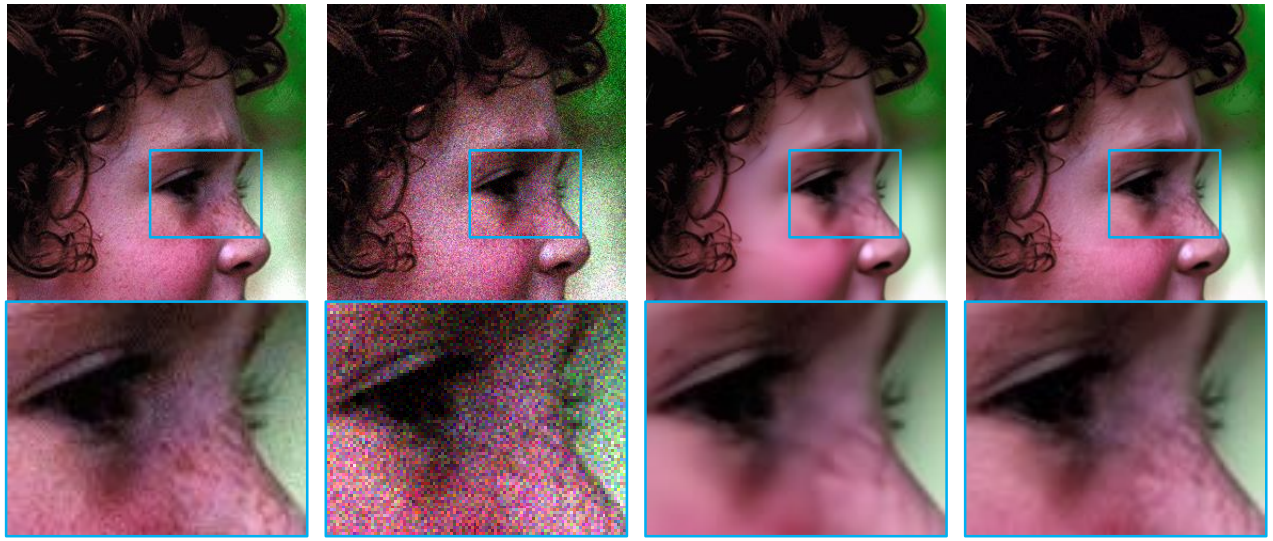
➔ Update P from the new estimate at each iteration

Poisson Denoising

Results

- Finer details recovered with the preconditioning
- In this case, preconditioning does not make the algorithm faster





Conclusion

Advantages:

- Preconditioning significantly improves the results of the plug-and-play ADMM for several applications
- Less iterations required
- Enables trading off universality for quality (i.e. fine tuning denoiser for specific noise level patterns)

Limitations:

- Impractical to train a universal denoiser for arbitrary noise covariance
- So far, only for diagonal preconditioning matrix ➔ suits problems with diagonal degradation matrix

Extending to applications with non-diagonal degradation matrix (e.g. deblurring/super-resolution)
➔ Requires other types of denoisers