Skoltech

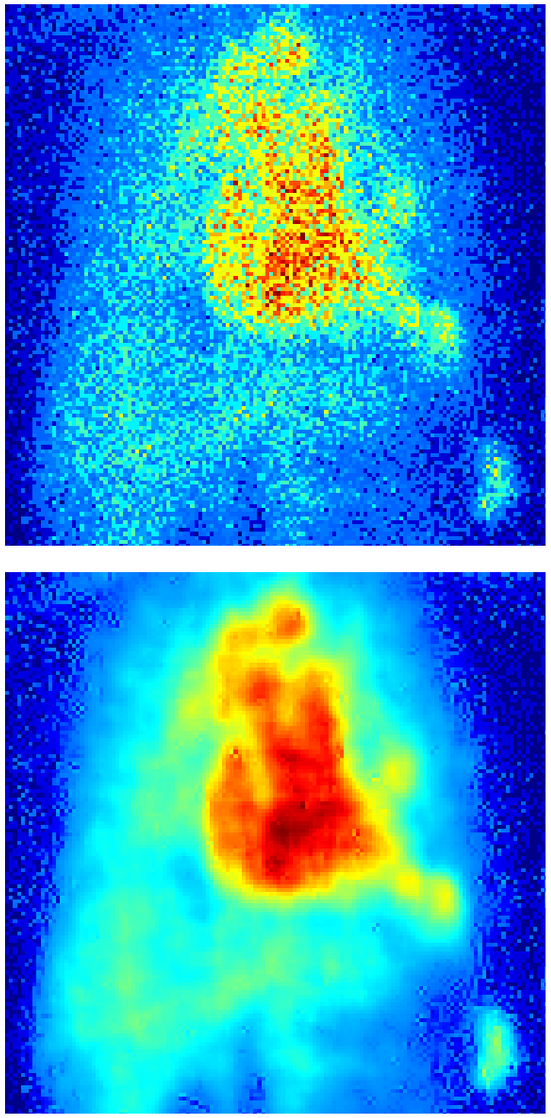
Skolkovo Institute of Science and Technology

Iterative Convolutional Neural Networks for Image Restoration

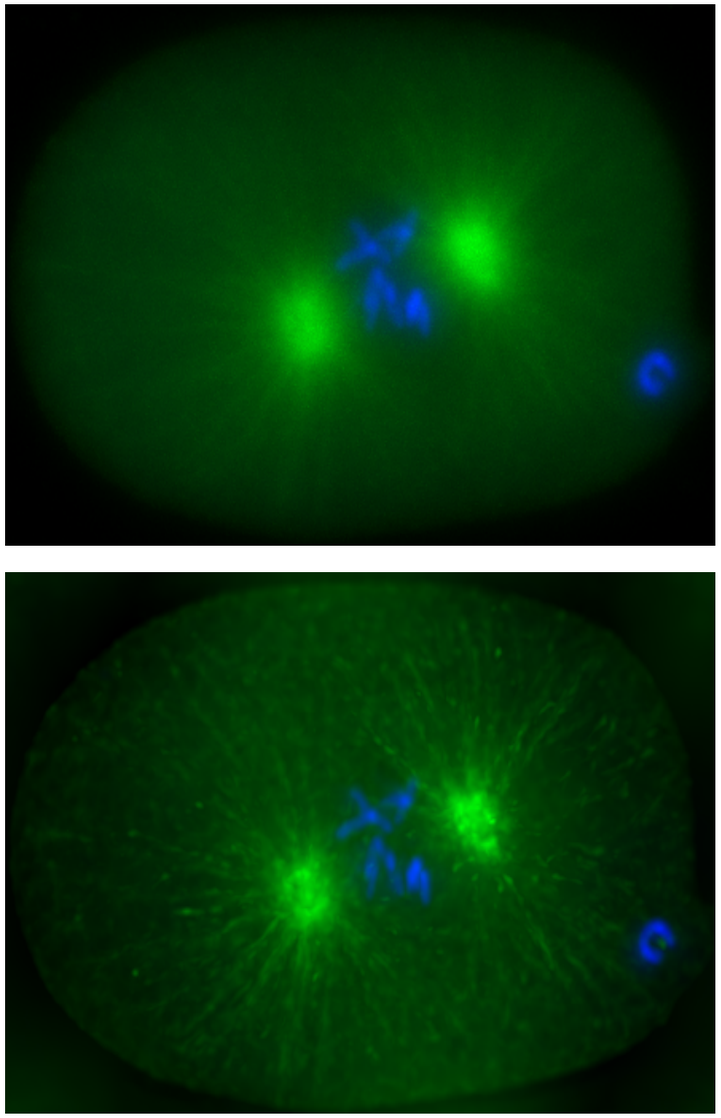
Stamatis Lefkimmiatis

Skolkovo Institute of Science and Technology

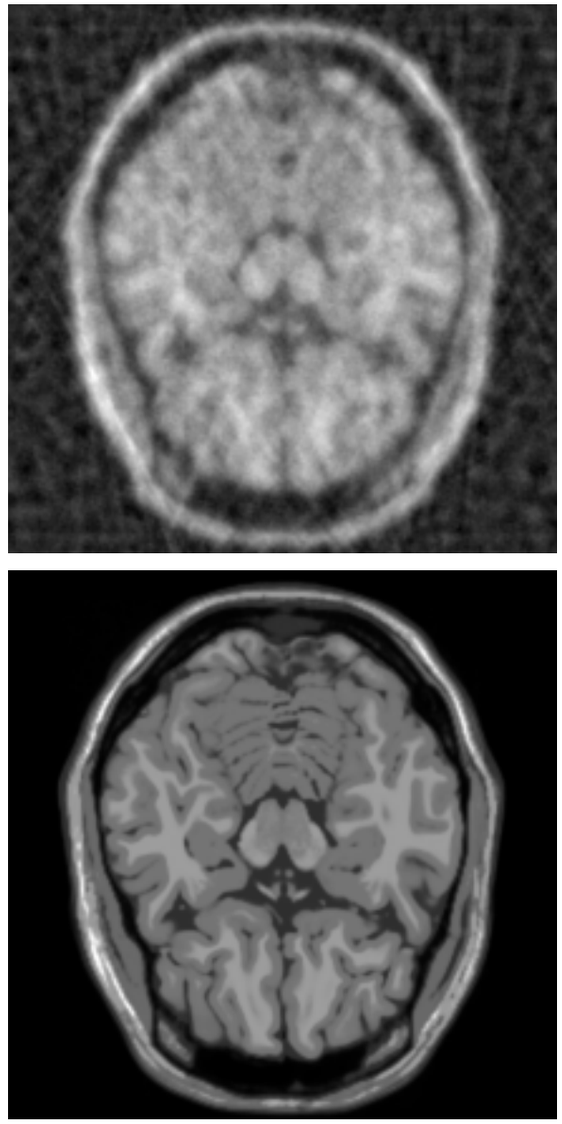
Inverse Problems in Image Processing



denoising



deblurring



MRI reconstruction

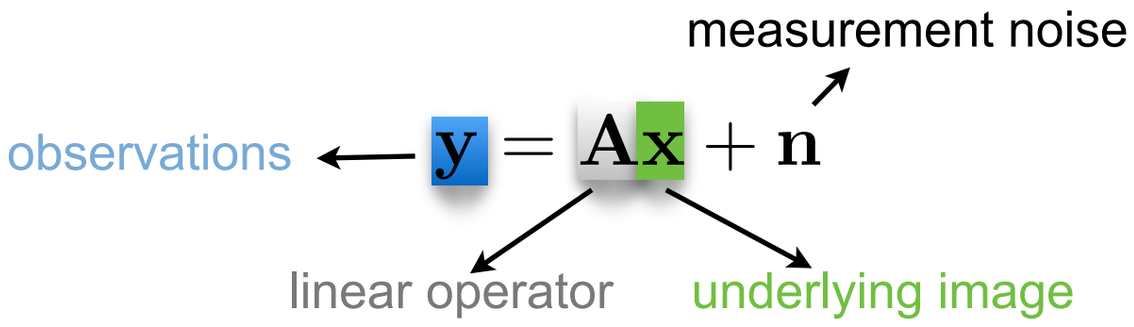
$\mathbf{y}$

$\mathbf{x}$

- Goal : recover $\mathbf{x}$ from a low-quality observed image $\mathbf{y}$

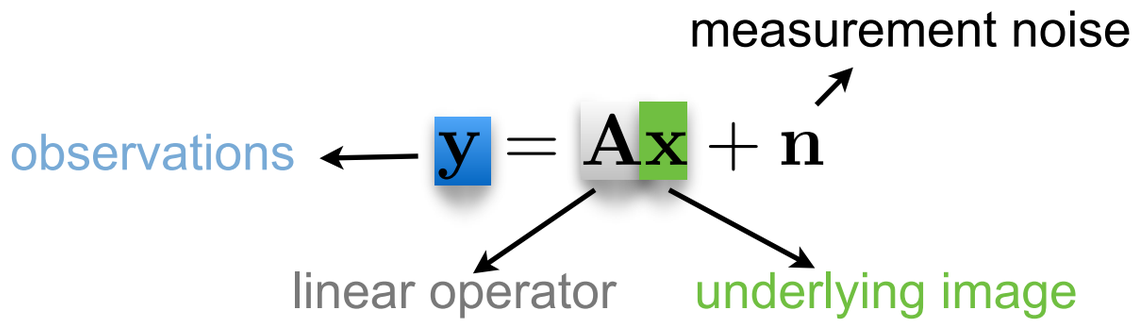
Variational / Bayesian Approach : Overview

- Forward model : Image acquisition can often be modeled as a linear process



Variational / Bayesian Approach : Overview

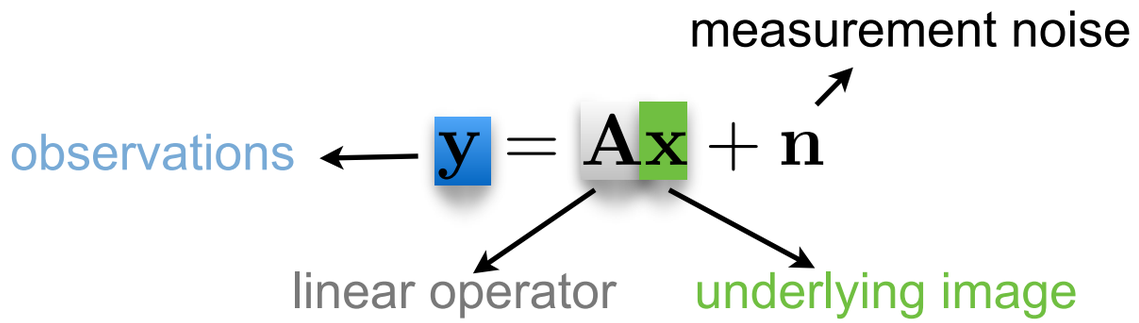
- Forward model : Image acquisition can often be modeled as a linear process



- Inverse Problems

Variational / Bayesian Approach : Overview

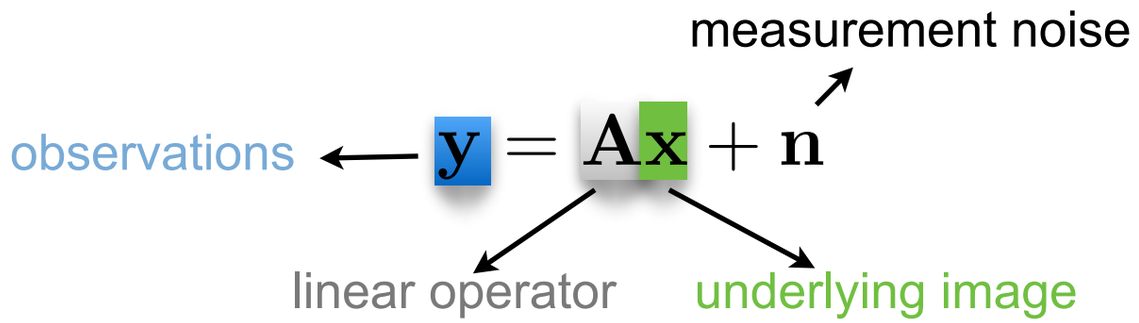
- Forward model : Image acquisition can often be modeled as a linear process



- Inverse Problems
  - **A** : Identity operator $\Rightarrow$ denoising

Variational / Bayesian Approach : Overview

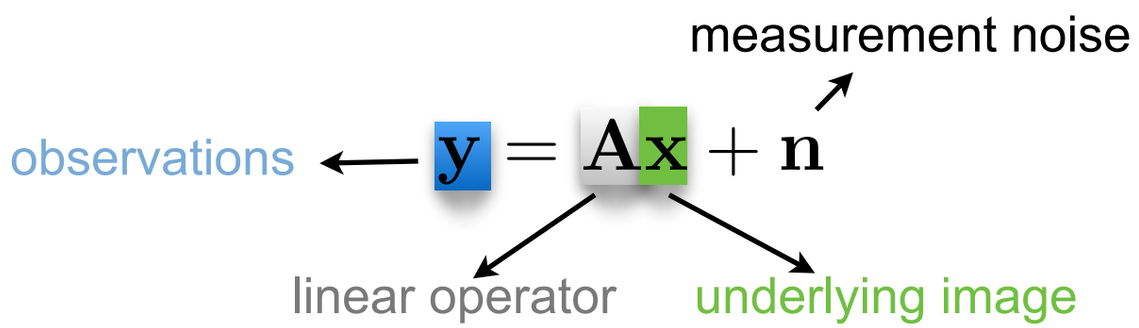
- Forward model : Image acquisition can often be modeled as a linear process



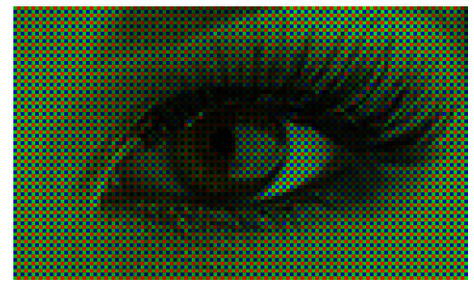
- Inverse Problems
  - **A** : Identity operator $\Rightarrow$ denoising
  - **A** : Convolutional operator $\Rightarrow$ deblurring

Variational / Bayesian Approach : Overview

- Forward model : Image acquisition can often be modeled as a linear process

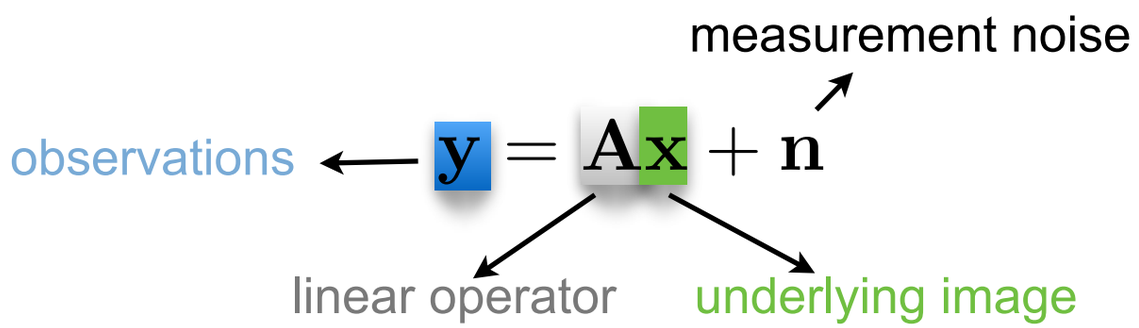


- Inverse Problems
  - **A** : Identity operator $\Rightarrow$ denoising
  - **A** : Convolutional operator $\Rightarrow$ deblurring
  - **A** : Binary mask $\Rightarrow$ inpainting, demosaicking

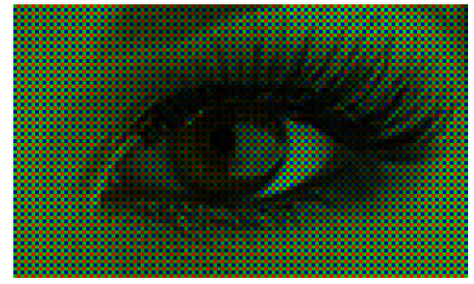


Variational / Bayesian Approach : Overview

- Forward model : Image acquisition can often be modeled as a linear process



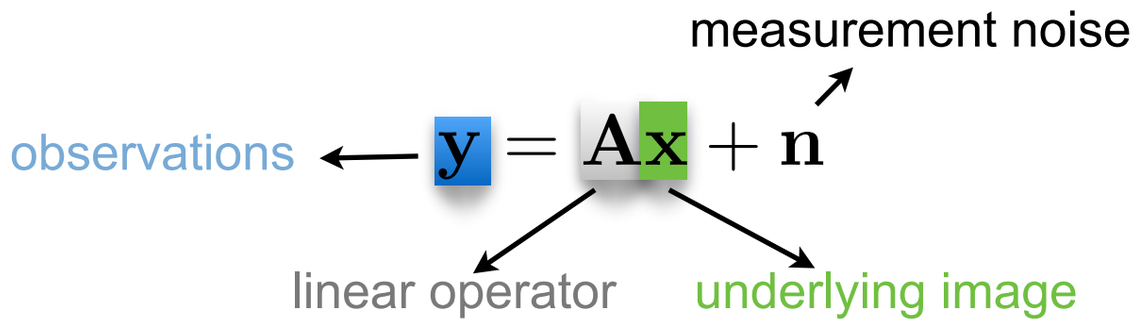
- Inverse Problems
  - **A** : Identity operator $\Rightarrow$ denoising
  - **A** : Convolutional operator $\Rightarrow$ deblurring
  - **A** : Binary mask $\Rightarrow$ inpainting, demosaicking
  - **A** : Smoothing + downsampling $\Rightarrow$ super-resolution



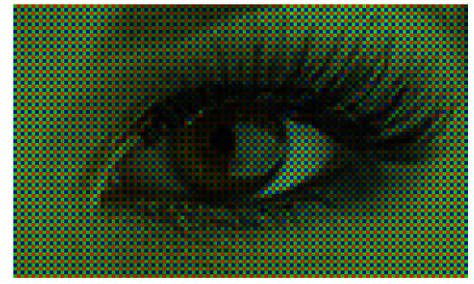



Variational / Bayesian Approach : Overview

- Forward model : Image acquisition can often be modeled as a linear process



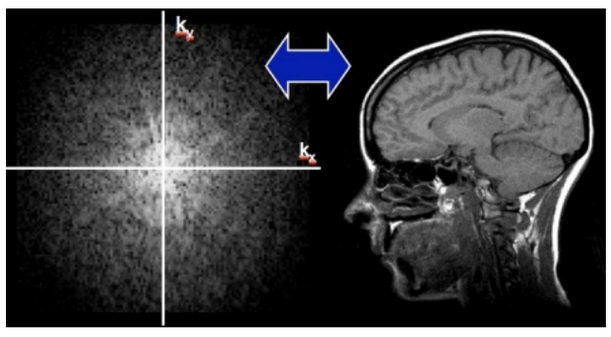
- Inverse Problems
  - **A** : Identity operator $\Rightarrow$ denoising
  - **A** : Convolutional operator $\Rightarrow$ deblurring
  - **A** : Binary mask $\Rightarrow$ inpainting, demosaicking
  - **A** : Smoothing + downsampling $\Rightarrow$ super-resolution
  - **A** : Fourier transform + binary mask $\Rightarrow$ MRI reconstruction



$\vdots$

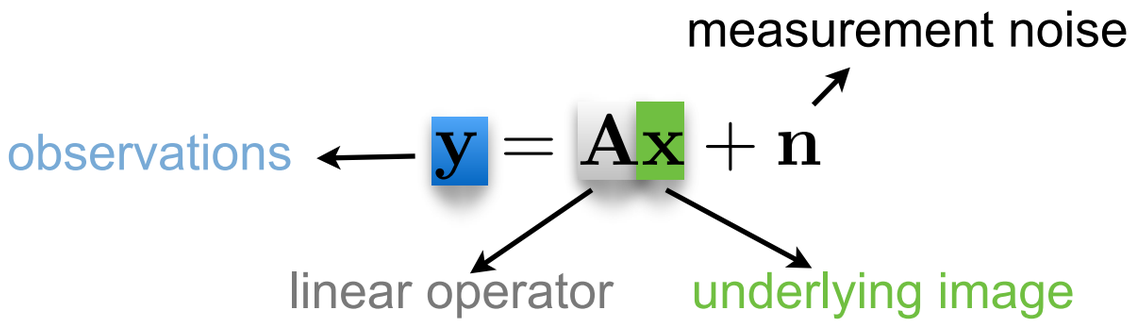






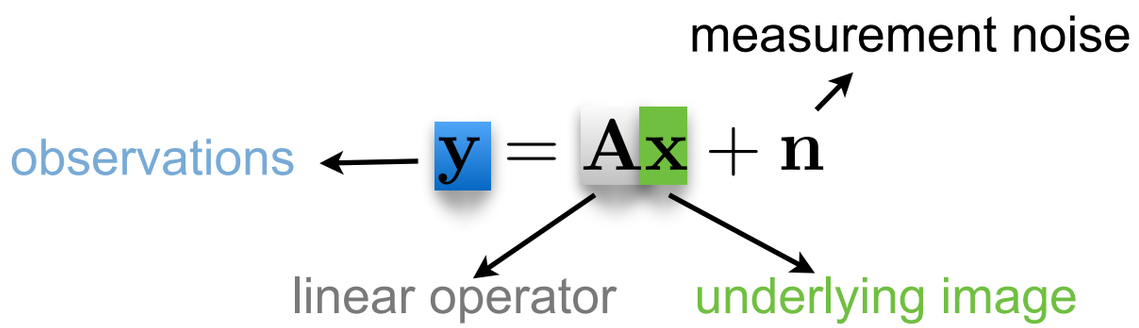
Variational / Bayesian Approach : Overview

- Forward model : Image acquisition can often be modeled as a linear process



Variational / Bayesian Approach : Overview

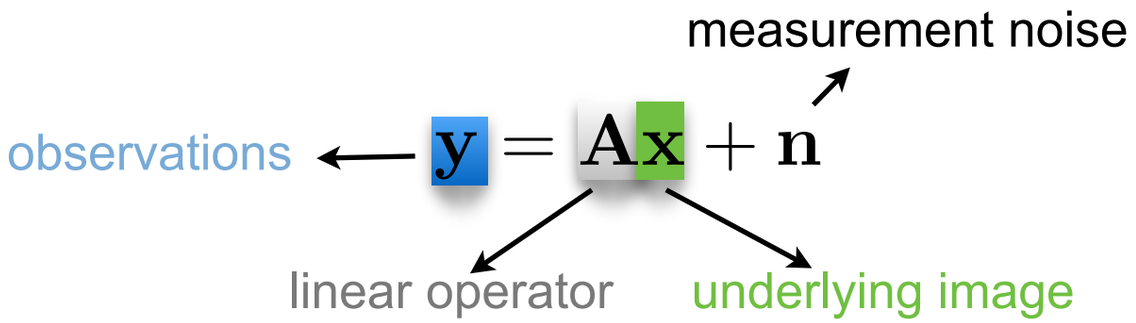
- Forward model : Image acquisition can often be modeled as a linear process



  - Image reconstruction
    - Ill-posed problem
    - Unique solution **does not exist**

Variational / Bayesian Approach : Overview

- Forward model : Image acquisition can often be modeled as a linear process



  - Image reconstruction
    - Ill-posed problem
    - Unique solution **does not exist**

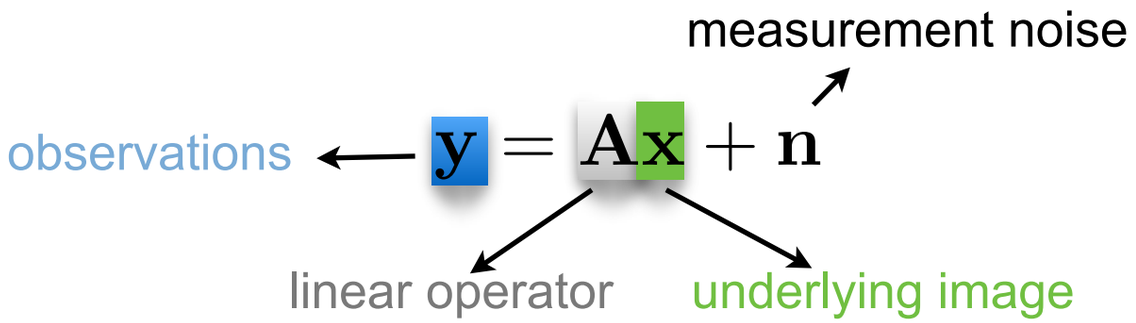
- Variational framework - Energy Minimization

data fidelity

$$E(\mathbf{x}) = d(\mathbf{x}; \mathbf{A}, \mathbf{y}) + \lambda \cdot r(\mathbf{x})$$ regularizer

Variational / Bayesian Approach : Overview

- Forward model : Image acquisition can often be modeled as a linear process



- Image reconstruction
  - Ill-posed problem
  - Unique solution **does not exist**
- Variational framework - Energy Minimization

$$E(\mathbf{x}) = \underbrace{d(\mathbf{x}; \mathbf{A}, \mathbf{y})}_{\text{data fidelity}} + \lambda \cdot \underbrace{r(\mathbf{x})}_{\text{regularizer}}$$

- Direct relation with Bayesian methods :
  - data fidelity $\Rightarrow$ observation log-likelihood
  - regularizer $\Rightarrow$ prior distribution
  - minimizer $\Rightarrow$ MAP estimate

Image Regularization

- Regularizer : key component of the reconstruction
  - Constrains the set of plausible solutions
  - Prior knowledge of desirable image properties

Image Regularization

- Regularizer : key component of the reconstruction
  - Constrains the set of plausible solutions
  - Prior knowledge of desirable image properties

- Generic form of regularization functionals :

$$r(\mathbf{x}) = \sum_{k=1}^{K} \phi(\mathbf{L}_k \mathbf{x})$$

- $\mathbf{L} : \mathbb{R}^N \mapsto \mathbb{R}^{K \times D}$**:** regularization operator (gradient, Hessian, wavelets, …)
- ϕ : potential function - ℓ_2 squared norm (Tikhonov-Miller regularization) , ℓ_1 norm, log function, ℓ_0 pseudo-norm, etc …

Image Regularization

- Regularizer : key component of the reconstruction
  - Constrains the set of plausible solutions
  - Prior knowledge of desirable image properties

- Generic form of regularization functionals :

$$r(\mathbf{x}) = \sum_{k=1}^{K} \phi(\mathbf{L}_k \mathbf{x})$$

  - $\mathbf{L} : \mathbb{R}^N \mapsto \mathbb{R}^{K \times D}$**:** regularization operator (gradient, Hessian, wavelets, …)
  - ϕ : potential function - ℓ_2 squared norm (Tikhonov-Miller regularization) , ℓ_1 norm, log function, ℓ_0 pseudo-norm, etc …

- Quadratic regularizers
  - mathematical simplicity
  - computational tractability
- Non-quadratic regularizers
  - less sensitive to outliers
  - improved reconstruction results

Total Variation

- One of the most popular regularizers for imaging applications [ROF `92]
  - Provides a global measure of the image intensity variations

Continuous-domain definition

$$\mathrm{TV}(u) = \int_{\Omega} \|\nabla u(\boldsymbol{r})\|_2 \, d\boldsymbol{r}$$

Discrete-domain definition

$$\mathrm{TV}(\mathbf{u}) = \sum_{n=1}^{N} \|(\nabla \mathbf{u})_n\|_2$$

- Numerous applications in image processing and computer vision
  - ✔ Invariance properties w.r.t transformations of the coordinate system
  - ✔ L_1 type of behavior that does not over-penalize large intensity variations
  - ✔ Reconstructed images with sharp edges
  - ✔ Convex functional - Efficient minimization techniques
  - ✗ Promotes piecewise constant reconstructions
  - ✗ Presence of staircase artifacts in smooth regions

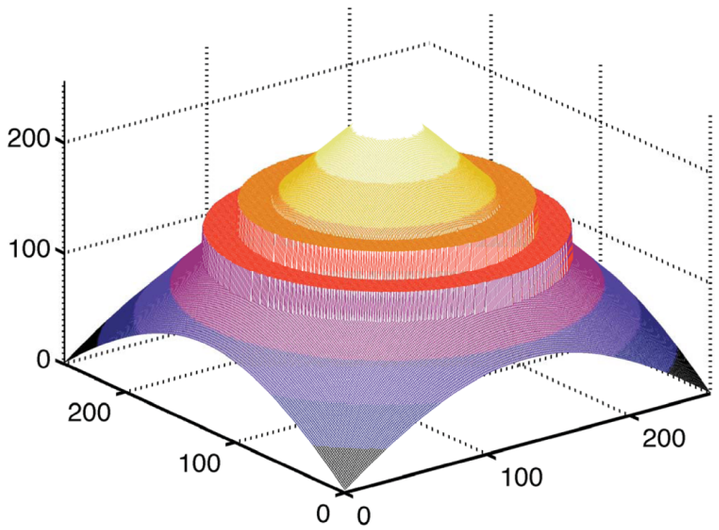
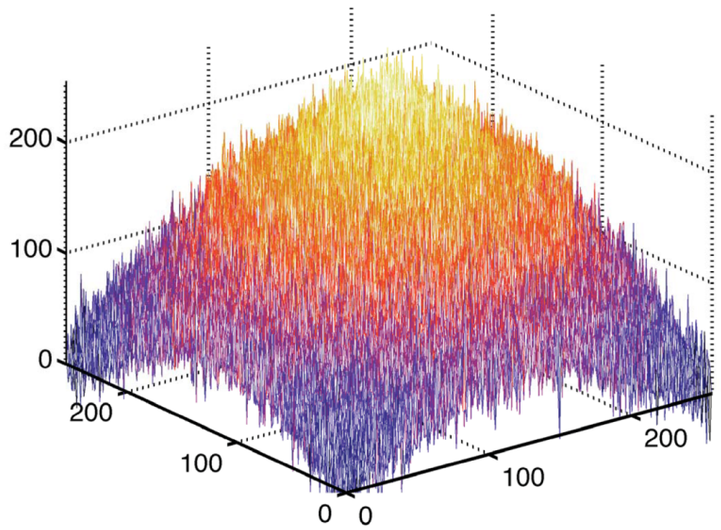
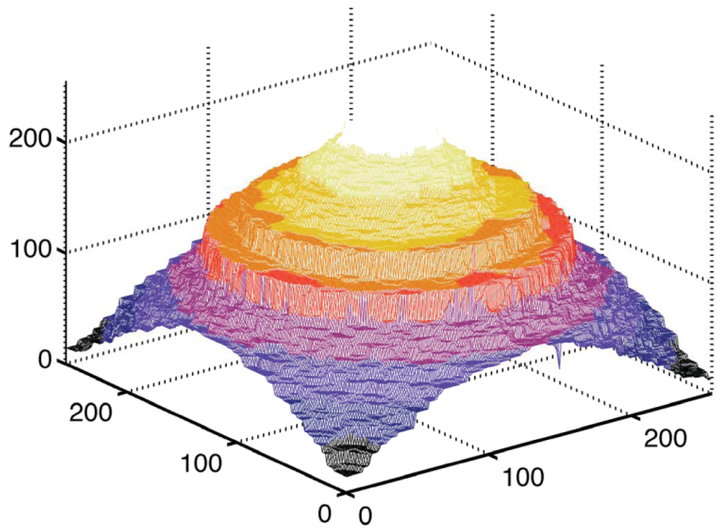


Image intensity profile



Noisy intensity profile



TV reconstruction

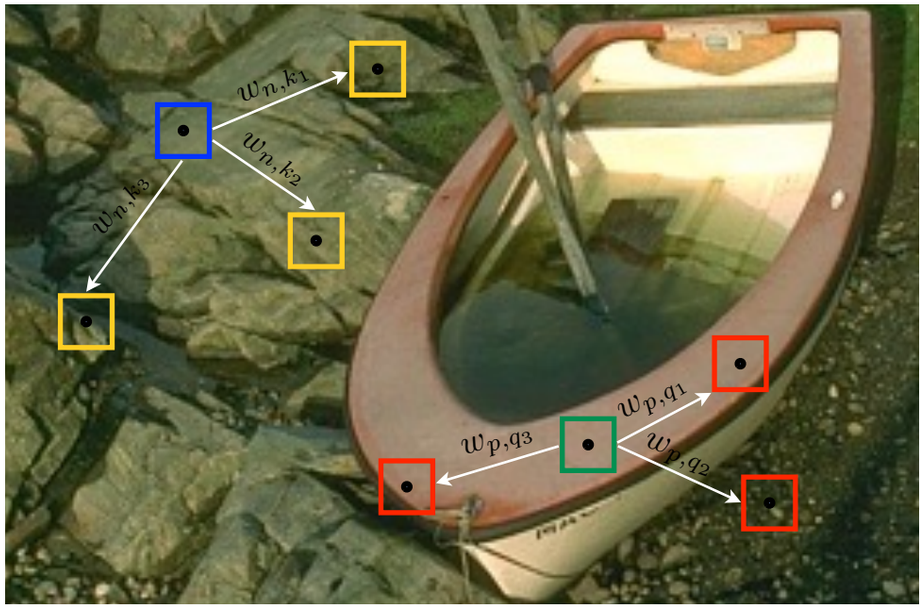
From Local to Non-Local Regularization

- TV-like regularizers only consider local features of the data
- Is there a way to capture more complex image structures?
  - Non-local Means : adaptive filter that considers non-local interactions [Buades et al. `05 & `10]

$$\mathrm{NL}(u)(\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \int_\Omega e^{-\frac{d_a(u(\boldsymbol{x}),u(\boldsymbol{y}))}{\beta^2}} u(\boldsymbol{y})\, d\boldsymbol{y}$$

patch-distance

$$d_a(u(\boldsymbol{x}), u(\boldsymbol{y})) = \int_\Omega G_a(t) \left|u(\boldsymbol{x}+\boldsymbol{t}) - u(\boldsymbol{y}+\boldsymbol{t})\right|^2 d\boldsymbol{t}$$



- Non-local TV : Variational extension of NL-means [Gilboa & Osher `08, Elmoataz et al. `08]
  - Key component : Non-local gradient on graphs

$$\nabla_w u(\boldsymbol{x}) = (u(\boldsymbol{y}) - u(\boldsymbol{x}))\sqrt{w(\boldsymbol{x},\boldsymbol{y})}, \quad \forall \boldsymbol{y} \in \Omega$$

non-local weight

- NL-TV : similar definition with classical TV
  - Important difference : image is defined on a graph

$$\mathrm{NLTV}(u) = \int_\Omega \|\nabla_w u(\boldsymbol{x})\|_2 \,\mathrm{d}\boldsymbol{x}$$

Machine Learning for Inverse Problems

- Two major challenges under the variational / minimization framework
  - Specify the exact form of the regularization operator and potential function
  - Recover the solution using an iterative approach which possibly requires a large number of updates

Machine Learning for Inverse Problems

- Two major challenges under the variational / minimization framework
  - Specify the exact form of the regularization operator and potential function
  - Recover the solution using an iterative approach which possibly requires a large number of updates
- Machine Learning alternative
  - Design a network with the capacity to learn the involved quantities from training data
  - Discriminative learning by using a loss-minimization strategy

Machine Learning for Inverse Problems

- Two major challenges under the variational / minimization framework
  - Specify the exact form of the regularization operator and potential function
  - Recover the solution using an iterative approach which possibly requires a large number of updates
- Machine Learning alternative
  - Design a network with the capacity to learn the involved quantities from training data
  - Discriminative learning by using a loss-minimization strategy
- Main challenge for ML
  - How to decide a proper architecture for the graph of the network?

Machine Learning for Inverse Problems

- Two major challenges under the variational / minimization framework
  - Specify the exact form of the regularization operator and potential function
  - Recover the solution using an iterative approach which possibly requires a large number of updates
- Machine Learning alternative
  - Design a network with the capacity to learn the involved quantities from training data
  - Discriminative learning by using a loss-minimization strategy
- Main challenge for ML
  - How to decide a proper architecture for the graph of the network?
- Principled approach for defining the network graph
  - Unroll an iterative optimization method
  - Use a limited number of updates to construct the network
  - Parametrize the regularization operator and the potential function

Optimization Strategy

- Recover the underlying image as the minimizer of the objective function

$$\arg\min_{\mathbf{x}} d\left(\mathbf{x}; \mathbf{A}, \mathbf{y}\right) + \lambda \cdot r\left(\mathbf{x}\right)$$

Optimization Strategy

- Recover the underlying image as the minimizer of the objective function

$$\arg\min_{\mathbf{x}} d\left(\mathbf{x}; \mathbf{A}, \mathbf{y}\right) + \lambda \cdot r\left(\mathbf{x}\right)$$

- Modern convex-optimization strategies for large-scale problems
  - FISTA (Majorization-Minimization) [Beck & Teboulle `09]
  - Split-Bregman / ADMM [Osher et al. `05, Goldstein & Osher `09, Esser `09, Afonso et al. `10 & `11, etc]
  - Primal-dual algorithms [Chambolle & Pock `10 & `11]
  - Proximal methods [Parikh & Boyd `13]

Optimization Strategy

- Recover the underlying image as the minimizer of the objective function

$$\arg\min_{\mathbf{x}} d\left(\mathbf{x}; \mathbf{A}, \mathbf{y}\right) + \lambda \cdot r\left(\mathbf{x}\right)$$

- Modern convex-optimization strategies for large-scale problems
  - FISTA (Majorization-Minimization) [Beck & Teboulle `09]
  - Split-Bregman / ADMM [Osher et al. `05, Goldstein & Osher `09, Esser `09, Afonso et al. `10 & `11, etc]
  - Primal-dual algorithms [Chambolle & Pock `10 & `11]
  - Proximal methods [Parikh & Boyd `13]

- Proximal map : building block for a number of optimization techniques

$$\text{prox}_{g(\cdot)}\left(\mathbf{z}\right) = \arg\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{2}\left\|\mathbf{x}-\mathbf{z}\right\|_2^2 + g\left(\mathbf{x}\right)$$

Optimization Strategy

- Recover the underlying image as the minimizer of the objective function

$$\arg\min_{\mathbf{x}} d\left(\mathbf{x}; \mathbf{A}, \mathbf{y}\right) + \lambda \cdot r\left(\mathbf{x}\right)$$

- Modern convex-optimization strategies for large-scale problems
  - FISTA (Majorization-Minimization) [Beck & Teboulle `09]
  - Split-Bregman / ADMM [Osher et al. `05, Goldstein & Osher `09, Esser `09, Afonso et al. `10 & `11, etc]
  - Primal-dual algorithms [Chambolle & Pock `10 & `11]
  - Proximal methods [Parikh & Boyd `13]

- Proximal map : building block for a number of optimization techniques

$$\text{prox}_{g(\cdot)}\left(\mathbf{z}\right) = \arg\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{2}\left\|\mathbf{x}-\mathbf{z}\right\|_2^2 + g\left(\mathbf{x}\right)$$

  - The proximal operator is a generalization of the orthogonal projection

Optimization Strategy

- Recover the underlying image as the minimizer of the objective function

$$\arg\min_{\mathbf{x}} d\left(\mathbf{x}; \mathbf{A}, \mathbf{y}\right) + \lambda \cdot r\left(\mathbf{x}\right)$$

- Modern convex-optimization strategies for large-scale problems
  - FISTA (Majorization-Minimization) [Beck & Teboulle `09]
  - Split-Bregman / ADMM [Osher et al. `05, Goldstein & Osher `09, Esser `09, Afonso et al. `10 & `11, etc]
  - Primal-dual algorithms [Chambolle & Pock `10 & `11]
  - Proximal methods [Parikh & Boyd `13]

- Proximal map : building block for a number of optimization techniques

$$\text{prox}_{g(\cdot)}\left(\mathbf{z}\right) = \arg\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{2}\left\|\mathbf{x}-\mathbf{z}\right\|_2^2 + g\left(\mathbf{x}\right)$$

  - The proximal operator is a generalization of the orthogonal projection
  - Can be interpreted as Gaussian denoising of $\mathbf{z}$ using $g\left(\mathbf{x}\right)$ as the regularizer

Image Denoising - Constrained Optimization

- Proximal map - image denoising solution (P1)

$$\mathbf{x}^{\star} = \arg\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right)$$

Image Denoising - Constrained Optimization

- Proximal map - image denoising solution (P1)

$$\mathbf{x}^{\star} = \arg\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right)$$

- Regularization parameter λ : “free” parameter
  - Different values lead to different restoration results of varying image quality
  - **No direct way** to *a priori* relate the value of λ with the restoration quality
  - Networks derived based on this formulation are noise-level specific [Schmidt & Roth `14, Chen et al. `16, Lefkimmiatis `17]
    - A new set of parameters’ for every noise level must be learned

Image Denoising - Constrained Optimization

- Proximal map - image denoising solution (P1)

$$\mathbf{x}^{\star} = \arg\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right)$$

- Regularization parameter λ : “free” parameter
  - Different values lead to different restoration results of varying image quality
  - **No direct way** to *a priori* relate the value of λ with the restoration quality
  - Networks derived based on this formulation are noise-level specific [Schmidt & Roth `14, Chen et al. `16, Lefkimmiatis `17]
    - A new set of parameters’ for every noise level must be learned
- Alternative way : Constrained problem formulation (P2)

$$\mathbf{x}^{\star} = \arg\min_{\|\mathbf{y}-\mathbf{x}\|_2 \leq \varepsilon} \left(\sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right) \equiv r\left(\mathbf{x}\right) \right)$$

  - $\|\mathbf{y} - \mathbf{x}\|_2 = \|\mathbf{n}\|_2 \propto \sigma$
  - Free parameter ε is easier to tune - directly related to the noise std
  - Available methods for estimating the noise level from observed data

Image Denoising - Constrained Optimization

- Proximal map - image denoising solution (P1)

$$\mathbf{x}^{\star} = \arg\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right)$$

- Regularization parameter λ : “free” parameter
  - Different values lead to different restoration results of varying image quality
  - **No direct way** to *a priori* relate the value of λ with the restoration quality
  - Networks derived based on this formulation are noise-level specific [Schmidt & Roth `14, Chen et al. `16, Lefkimmiatis `17]
    - A new set of parameters’ for every noise level must be learned
- Alternative way : Constrained problem formulation (P2)

$$\mathbf{x}^{\star} = \underset{\|\mathbf{y}-\mathbf{x}\|_2 \leq \varepsilon}{\arg\min} \left(\sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right) \equiv r\left(\mathbf{x}\right) \right)$$

  - $\|\mathbf{y} - \mathbf{x}\|_2 = \|\mathbf{n}\|_2 \propto \sigma$
  - Free parameter ε is easier to tune - directly related to the noise std
  - Available methods for estimating the noise level from observed data
- The two problems are equivalent in the following sense :
  - For any $\varepsilon > 0$ such that P2 is feasible, the solution of P2 is either the null vector or else it is a solution of P1 for some $\lambda > 0$

Proximal Gradient Method

- Unconstrained form of P2

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right) + \iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right)$$

indicator function

$$\iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right) = \begin{cases} 0, & \text{if } \|\mathbf{y}-\mathbf{x}\|_2 \le \varepsilon \\ \infty, & \text{otherwise} \end{cases}$$

Proximal Gradient Method

- Unconstrained form of P2

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right) + \iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right)$$

$$\text{indicator function} \quad \iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right) = \begin{cases} 0, & \text{if } \|\mathbf{y}-\mathbf{x}\|_2 \leq \varepsilon \\ \infty, & \text{otherwise} \end{cases}$$

- PGM : gradient-descent method that can handle non-smooth functionals

Proximal Gradient Method

- Unconstrained form of P2

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right) + \iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right)$$

$$\text{indicator function} \quad \iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right) = \begin{cases} 0, & \text{if } \|\mathbf{y}-\mathbf{x}\|_2 \leq \varepsilon \\ \infty, & \text{otherwise} \end{cases}$$

- PGM : gradient-descent method that can handle non-smooth functionals
  - The objective is split into a smooth and a non-smooth part
  - Must be able to compute the proximal operator of the non-smooth functional

$$\text{prox}_{\iota_{\mathcal{C}(\mathbf{y},\varepsilon)}}\left(\mathbf{z}\right) = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + \iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right)$$

Proximal Gradient Method

- Unconstrained form of P2

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right) + \iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right)$$

indicator function

$$\iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right) = \begin{cases} 0, & \text{if } \|\mathbf{y}-\mathbf{x}\|_2 \leq \varepsilon \\ \infty, & \text{otherwise} \end{cases}$$

- PGM : gradient-descent method that can handle non-smooth functionals
  - The objective is split into a smooth and a non-smooth part
  - Must be able to compute the proximal operator of the non-smooth functional

$$\text{prox}_{\iota_{\mathcal{C}(\mathbf{y},\varepsilon)}}\left(\mathbf{z}\right) = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + \iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right)$$

- Updates of the solution using PGM (assuming a smooth regularizer)

adaptive step size

$$\mathbf{x}^t = \text{prox}_{\gamma^t \iota_C}\left(\mathbf{x}^{t-1} - \gamma^t \nabla_{\mathbf{x}} r\left(\mathbf{x}^{t-1}\right)\right)$$

Proximal Gradient Method Cont'd

- Gradient of the smooth part

$$\nabla_{\mathbf{x}} r(\mathbf{x}) = \sum_{k=1}^{K} \mathbf{L}_k^{\mathsf{T}} \psi(\mathbf{L}_k \mathbf{x}) \equiv h(\mathbf{x})$$

$$\psi(\mathbf{z}) = \nabla_{\mathbf{z}} \phi(\mathbf{z}), \ \mathbf{z} \in \mathbb{R}^D$$

Proximal Gradient Method Cont'd

- Gradient of the smooth part

$$\nabla_{\mathbf{x}} r(\mathbf{x}) = \sum_{k=1}^{K} \mathbf{L}_k^{\mathsf{T}} \psi(\mathbf{L}_k \mathbf{x}) \equiv h(\mathbf{x})$$

$$\psi(\mathbf{z}) = \nabla_{\mathbf{z}} \phi(\mathbf{z}), \ \mathbf{z} \in \mathbb{R}^D$$

- Proximal map of the non-smooth part
  - Orthogonal projection onto the set $\mathcal{C}(\mathbf{y}, \varepsilon)$

$$\Pi_{\mathcal{C}}(\mathbf{z}) = \underset{\mathbf{x} \in \mathcal{C}(\mathbf{y}, \varepsilon)}{\arg\min} \|\mathbf{x} - \mathbf{z}\|_2^2$$

Proximal Gradient Method Cont'd

- Gradient of the smooth part

$$\nabla_{\mathbf{x}} r(\mathbf{x}) = \sum_{k=1}^{K} \mathbf{L}_k^{\mathsf{T}} \psi(\mathbf{L}_k \mathbf{x}) \equiv h(\mathbf{x})$$

$$\psi(\mathbf{z}) = \nabla_{\mathbf{z}} \phi(\mathbf{z}), \ \mathbf{z} \in \mathbb{R}^D$$

- Proximal map of the non-smooth part
  - Orthogonal projection onto the set $\mathcal{C}(\mathbf{y}, \varepsilon)$

$$\Pi_{\mathcal{C}}(\mathbf{z}) = \underset{\mathbf{x} \in \mathcal{C}(\mathbf{y}, \varepsilon)}{\arg\min} \|\mathbf{x} - \mathbf{z}\|_2^2$$

  - An analytical solution **exists**

$$\Pi_{\mathcal{C}}(\boldsymbol{z}) = \mathbf{y} + \varepsilon \frac{\boldsymbol{z} - \mathbf{y}}{\max(\|\boldsymbol{z} - \mathbf{y}\|_2, \varepsilon)}$$

Proximal Gradient Method Cont'd

- Gradient of the smooth part

$$\nabla_{\mathbf{x}} r(\mathbf{x}) = \sum_{k=1}^{K} \mathbf{L}_k^{\mathsf{T}} \psi(\mathbf{L}_k \mathbf{x}) \equiv h(\mathbf{x})$$

$$\psi(\mathbf{z}) = \nabla_{\mathbf{z}} \phi(\mathbf{z}), \ \mathbf{z} \in \mathbb{R}^D$$

- Proximal map of the non-smooth part
  - Orthogonal projection onto the set $\mathcal{C}(\mathbf{y}, \varepsilon)$

$$\Pi_{\mathcal{C}}(\mathbf{z}) = \underset{\mathbf{x} \in \mathcal{C}(\mathbf{y}, \varepsilon)}{\arg\min} \|\mathbf{x} - \mathbf{z}\|_2^2$$

  - An analytical solution **exists**

$$\Pi_{\mathcal{C}}(\boldsymbol{z}) = \mathbf{y} + \varepsilon \frac{\boldsymbol{z} - \mathbf{y}}{\max(\|\boldsymbol{z} - \mathbf{y}\|_2, \varepsilon)}$$

- Proximal gradient iterations

$$\mathbf{x}^t = \Pi_{\mathcal{C}}\left(\mathbf{x}^{t-1} - h^t\left(\mathbf{x}^{t-1}\right)\right), \text{ where } h^t(\mathbf{x}) = \gamma^t h(\mathbf{x})$$

Normalized residual iterations

- PGM updates

$$\mathbf{x}^t = \Pi_{\mathcal{C}} \left(\mathbf{x}^{t-1} - h^t \left(\mathbf{x}^{t-1} \right) \right), \text{ where } h^t \left(\mathbf{x} \right) = \gamma^t \, h \left(\mathbf{x} \right)$$

Normalized residual iterations

- PGM updates

$$\mathbf{x}^t = \Pi_{\mathcal{C}} \left(\mathbf{x}^{t-1} - h^t \left(\mathbf{x}^{t-1} \right) \right), \text{ where } h^t \left(\mathbf{x} \right) = \gamma^t \, h \left(\mathbf{x} \right)$$

- Interpretation : Iterative residual denoising
  - The solution is obtained by recursively subtracting from the input refined estimates of the noise realization distorting the input

Normalized residual iterations

- PGM updates

$$\mathbf{x}^t = \Pi_{\mathcal{C}} \left(\mathbf{x}^{t-1} - h^t \left(\mathbf{x}^{t-1} \right) \right), \text{ where } h^t \left(\mathbf{x} \right) = \gamma^t \, h \left(\mathbf{x} \right)$$

- Interpretation : Iterative residual denoising
  - The solution is obtained by recursively subtracting from the input refined estimates of the noise realization distorting the input

$$\mathbf{x}^1 = \mathbf{y} - \varepsilon \frac{h^1 \left(\mathbf{y} \right)}{\max \left(\left\| h^1 \left(\mathbf{y} \right) \right\|_2, \varepsilon \right)} = \mathbf{x} + \left(\mathbf{n} - \mathbf{n}^1 \right)$$

Normalized residual iterations

- PGM updates

$$\mathbf{x}^t = \Pi_{\mathcal{C}}\left(\mathbf{x}^{t-1} - h^t\left(\mathbf{x}^{t-1}\right)\right), \text{ where } h^t(\mathbf{x}) = \gamma^t h(\mathbf{x})$$

- Interpretation : Iterative residual denoising
  - The solution is obtained by recursively subtracting from the input refined estimates of the noise realization distorting the input

$$\mathbf{x}^1 = \mathbf{y} - \varepsilon \frac{h^1(\mathbf{y})}{\max\left(\left\|h^1(\mathbf{y})\right\|_2, \varepsilon\right)} = \mathbf{x} + \left(\mathbf{n} - \mathbf{n}^1\right)$$

$$\vdots$$

$$\mathbf{x}^k = \mathbf{y} - \mathbf{n}^k = \mathbf{x} + \left(\mathbf{n} - \mathbf{n}^k\right), \quad k > 1$$

Normalized residual iterations

- PGM updates

$$\mathbf{x}^t = \Pi_{\mathcal{C}} \left(\mathbf{x}^{t-1} - h^t \left(\mathbf{x}^{t-1} \right) \right), \text{ where } h^t (\mathbf{x}) = \gamma^t \, h (\mathbf{x})$$

- Interpretation : Iterative residual denoising
  - The solution is obtained by recursively subtracting from the input refined estimates of the noise realization distorting the input

$$\mathbf{x}^1 = \mathbf{y} - \varepsilon \frac{h^1 (\mathbf{y})}{\max \left(\left\| h^1 (\mathbf{y}) \right\|_2, \varepsilon \right)} = \mathbf{x} + \left(\mathbf{n} - \mathbf{n}^1 \right)$$

$$\vdots$$

$$\mathbf{x}^k = \mathbf{y} - \mathbf{n}^k = \mathbf{x} + \left(\mathbf{n} - \mathbf{n}^k \right), \quad k > 1$$

$$\mathbf{n}^k = \varepsilon \frac{\mathbf{n}^{k-1} + h^k \left(\mathbf{x}^{k-1} \right)}{\max \left(\left\| \mathbf{n}^{k-1} + h^k \left(\mathbf{x}^{k-1} \right) \right\|_2, \varepsilon \right)}$$

normalized noise realization estimate

Universal Network Architecture

- Cascade of composite layers (“stages”)
- Each layer implements a single PGM update

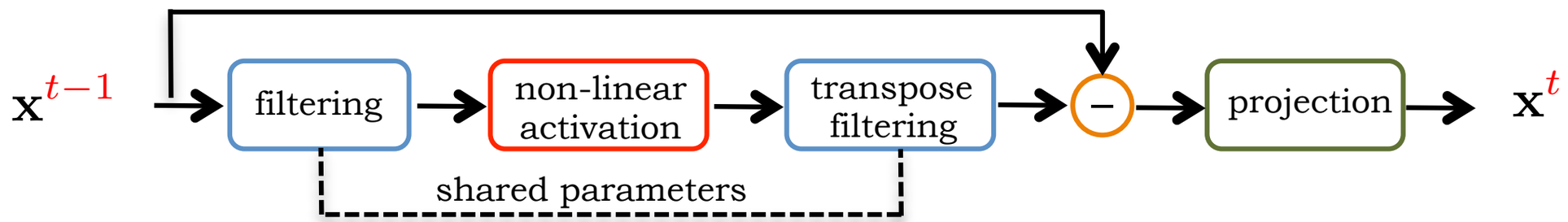
$$\mathbf{x}^t = \Pi_{\mathcal{C}} \left(\mathbf{x}^{t-1} - \sum_{k=1}^{K} \mathbf{L}_k^{\mathsf{T}} \psi^t \left(\mathbf{L}_k \mathbf{x}^{t-1} \right) \right)$$

Universal Network Architecture

- Cascade of composite layers (“stages”)
- Each layer implements a single PGM update

$$\mathbf{x}^t = \Pi_{\mathcal{C}} \left(\mathbf{x}^{t-1} - \sum_{k=1}^{K} \mathbf{L}_k^{\mathsf{T}} \psi^t \left(\mathbf{L}_k \mathbf{x}^{t-1} \right) \right)$$

$$\Downarrow$$

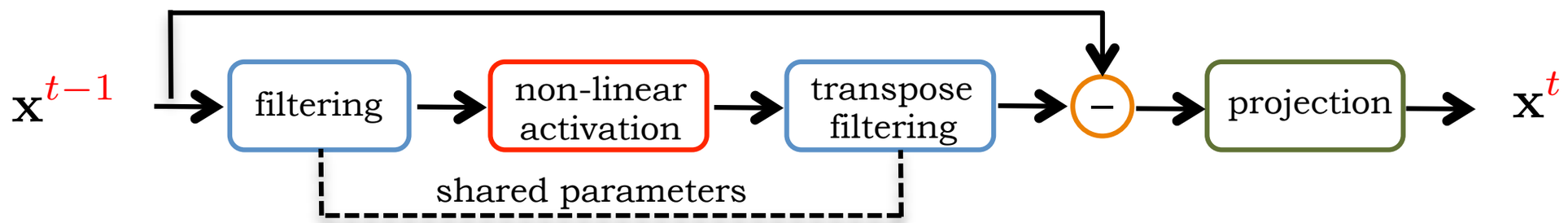


Universal Network Architecture

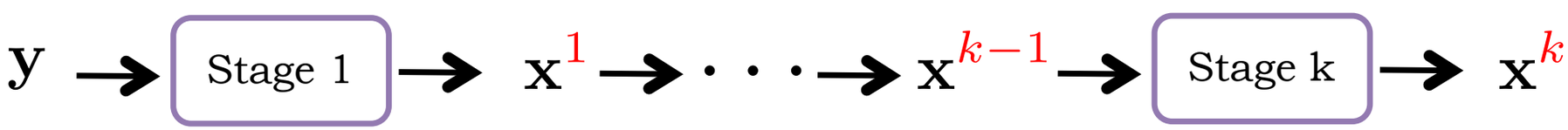
- Cascade of composite layers (“stages”)
- Each layer implements a single PGM update

$$\mathbf{x}^t = \Pi_{\mathcal{C}} \left(\mathbf{x}^{t-1} - \sum_{k=1}^{K} \mathbf{L}_k^{\mathsf{T}} \psi^t \left(\mathbf{L}_k \mathbf{x}^{t-1} \right) \right)$$

$\Downarrow$



- Feedforward network



Local Deep Network

- Regularization operator $\mathbf{L} : \mathbb{R}^N \mapsto \mathbb{R}^{K \times D}$
  - Typical choices are differential operators or wavelets
  - Both can be implemented as filter-banks of high-pass filters
  - In deep learning terminology these are “convolutional layers”

Local Deep Network

- Regularization operator $\mathbf{L} : \mathbb{R}^N \mapsto \mathbb{R}^{K \times D}$
  - Typical choices are differential operators or wavelets
  - Both can be implemented as filter-banks of high-pass filters
  - In deep learning terminology these are “convolutional layers”

- Normalized convolution
  - To derive valid regularizers the learned filters must be zero-mean
  - To speed-up training and to make sure that different pairs of potential functions and filters result in different loss we further impose that the learned filters have fixed scale

parametric form of filter weights

mean value

$$\boldsymbol{w} = s \frac{\mathbf{v} - \bar{\mathbf{v}}}{\|\mathbf{v} - \bar{\mathbf{v}}\|} \in \mathbb{R}^L$$

Local Deep Network

- Regularization operator $\mathbf{L} : \mathbb{R}^N \mapsto \mathbb{R}^{K \times D}$
  - Typical choices are differential operators or wavelets
  - Both can be implemented as filter-banks of high-pass filters
  - In deep learning terminology these are “convolutional layers”

- Normalized convolution
  - To derive valid regularizers the learned filters must be zero-mean
  - To speed-up training and to make sure that different pairs of potential functions and filters result in different loss we further impose that the learned filters have fixed scale

parametric form of filter weights

mean value

$$\boldsymbol{w} = s \frac{\mathbf{v} - \bar{\mathbf{v}}}{\|\mathbf{v} - \bar{\mathbf{v}}\|} \in \mathbb{R}^L$$

- Similar parametrization has been used in other deep networks for classification tasks to replace batch-normalization [Salimans & Kingma `16]

Potential Function Parameterization

- Variational Framework
  - Potential function : important role for the final quality of reconstruction
    - Common choices: ℓ_p norms, log function, ℓ_0 pseudo-norm, etc

Potential Function Parameterization

- Variational Framework
  - Potential function : important role for the final quality of reconstruction
    - Common choices: ℓ_p norms, log function, ℓ_0 pseudo-norm, etc
- ML Framework
  - Goal : rich and expressive parameterization for the potential function
  - main assumption : potential function is **smooth** and **separable**

$$\phi(\mathbf{z}) = \sum_{d=1}^{D} \phi_d(\mathbf{z}_d), \quad \mathbf{z} \in \mathbb{R}^D$$

Potential Function Parameterization

- Variational Framework
  - Potential function : important role for the final quality of reconstruction
    - Common choices: ℓ_p norms, log function, ℓ_0 pseudo-norm, etc
- ML Framework
  - Goal : rich and expressive parameterization for the potential function
  - main assumption : potential function is **smooth** and **separable**

$$\phi(\mathbf{z}) = \sum_{d=1}^{D} \phi_d(\mathbf{z}_d), \quad \mathbf{z} \in \mathbb{R}^D$$

  - PGM updates reminder : derivative of the potential function

$$\nabla_{\mathbf{x}} r(\mathbf{x}) = \sum_{k=1}^{K} \mathbf{L}_k^{\mathsf{T}} \psi(\mathbf{L}_k \mathbf{x}) \equiv h(\mathbf{x})$$

$$\psi(\mathbf{z}) = \begin{bmatrix} \psi_1(\mathbf{z}_1) & \psi_2(\mathbf{z}_2) & \dots & \psi_D(\mathbf{z}_D) \end{bmatrix}^{\mathsf{T}}, \quad \psi_d(z) = \frac{d\phi_d}{dz}(z)$$

Potential Function Parameterization

- Variational Framework
  - Potential function : important role for the final quality of reconstruction
    - Common choices: ℓ_p norms, log function, ℓ_0 pseudo-norm, etc
- ML Framework
  - Goal : rich and expressive parameterization for the potential function
  - main assumption : potential function is **smooth** and **separable**

$$\phi(\mathbf{z}) = \sum_{d=1}^{D} \phi_d(\mathbf{z}_d), \quad \mathbf{z} \in \mathbb{R}^D$$

  - PGM updates reminder : derivative of the potential function

$$\nabla_{\mathbf{x}} r(\mathbf{x}) = \sum_{k=1}^{K} \mathbf{L}_k^{\mathsf{T}} \psi(\mathbf{L}_k \mathbf{x}) \equiv h(\mathbf{x})$$

$$\psi(\mathbf{z}) = \begin{bmatrix} \psi_1(\mathbf{z}_1) & \psi_2(\mathbf{z}_2) & \dots & \psi_D(\mathbf{z}_D) \end{bmatrix}^{\mathsf{T}}, \quad \psi_d(z) = \frac{d\phi_d}{dz}(z)$$

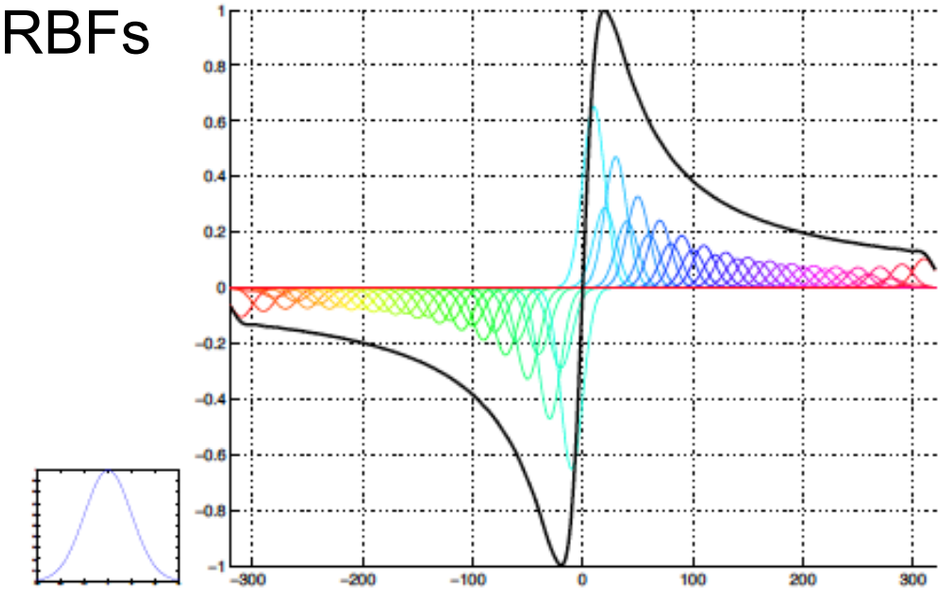
  - Indirect modeling using linear combination of RBFs

$$\psi_i(x) = \sum_{j=1}^{M} \pi_{ij} \rho_j(|x - \mu_j|)$$

π_{ij} : expansion coefficients

μ_j : centers of basis functions

$\rho_j = \exp\left(-\varepsilon_j r^2\right)$: Gaussian radial functions



Non-local Deep Network

- Non-local regularization methods lead to superior reconstruction results
  - Allow long-range dependencies between points in the image domain
  - Ability to model complex image structures

Non-local Deep Network

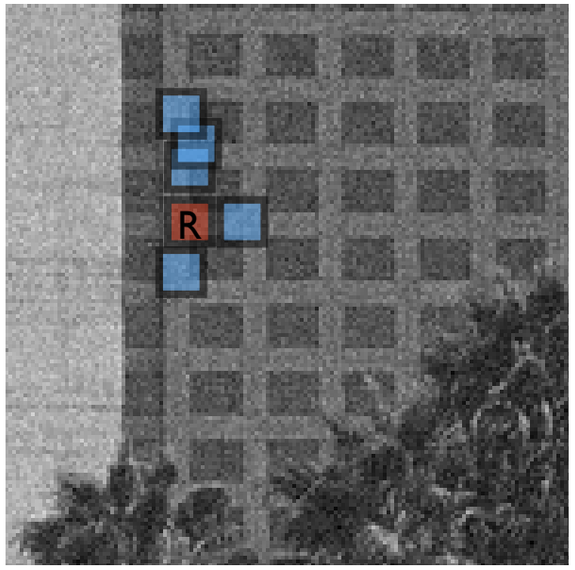
- Non-local regularization methods lead to superior reconstruction results
  - Allow long-range dependencies between points in the image domain
  - Ability to model complex image structures

- Non-local Regularization Operator
  - Core component of our non-local deep network
  - Inspiration by BM3D algorithm [Dabov et. al `07]
  - Three-step operation

Non-local Deep Network

- Non-local regularization methods lead to superior reconstruction results
  - Allow long-range dependencies between points in the image domain
  - Ability to model complex image structures

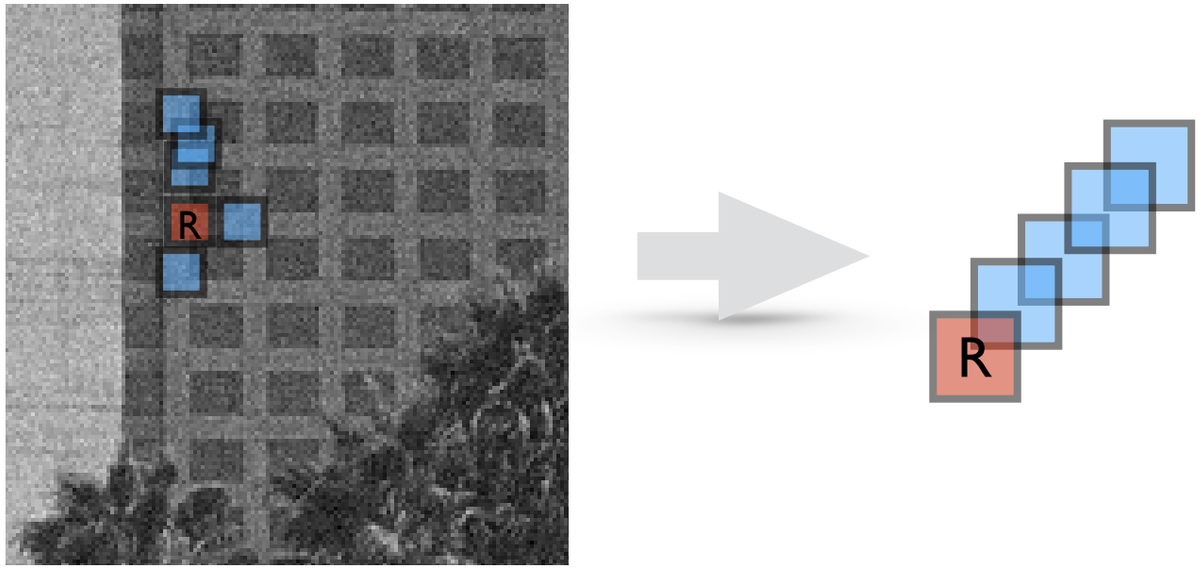
- Non-local Regularization Operator
  - Core component of our non-local deep network
  - Inspiration by BM3D algorithm [Dabov et. al `07]
  - Three-step operation



Block-Matching

Non-local Deep Network

- Non-local regularization methods lead to superior reconstruction results
  - Allow long-range dependencies between points in the image domain
  - Ability to model complex image structures
- Non-local Regularization Operator
  - Core component of our non-local deep network
  - Inspiration by BM3D algorithm [Dabov et. al `07]
  - Three-step operation



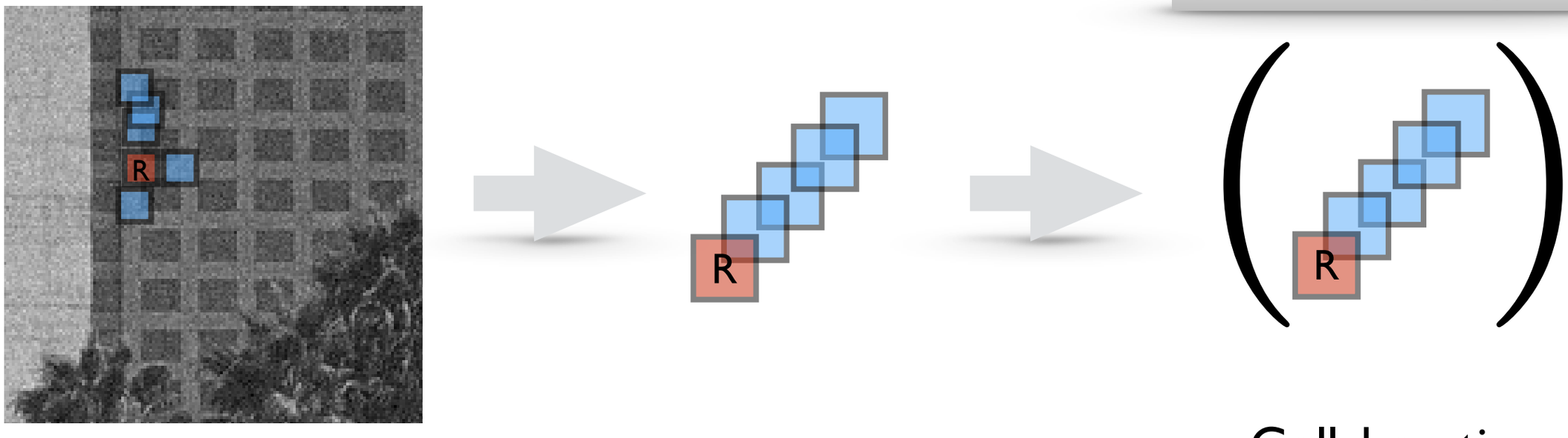
Block-Matching

Grouping

Non-local Deep Network

- Non-local regularization methods lead to superior reconstruction results
  - Allow long-range dependencies between points in the image domain
  - Ability to model complex image structures

- Non-local Regularization Operator
  - Core component of our non-local deep network
  - Inspiration by BM3D algorithm [Dabov et. al `07]
  - Three-step operation



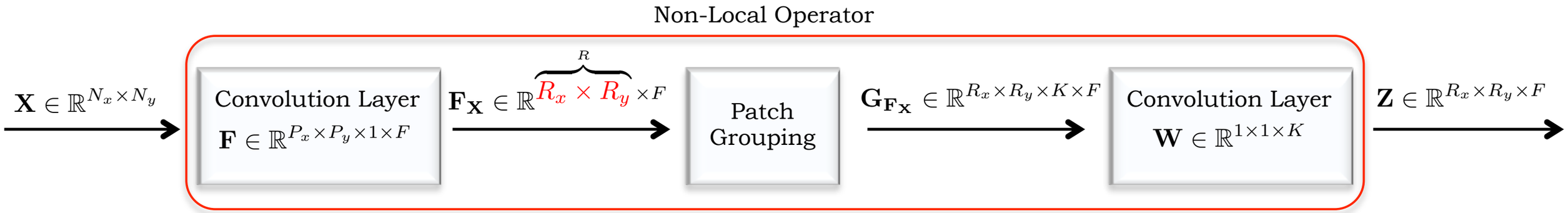
Block-Matching

Grouping

Collaborative Filtering

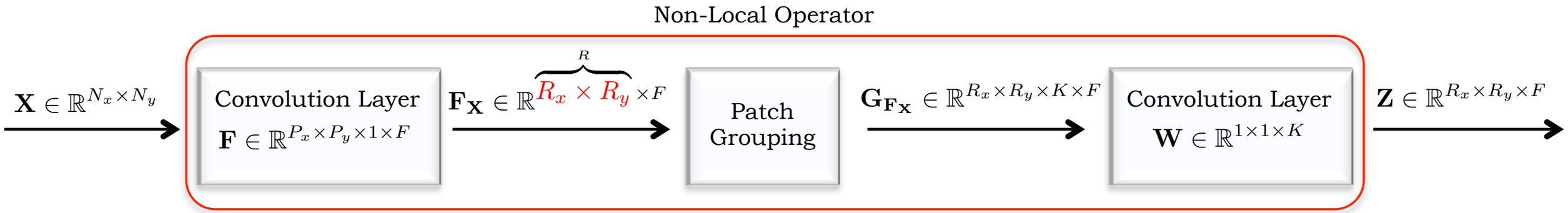
Non-local Regularization Operator

- Construction of the Non-Local operator
  - Patch extraction & patch-transform can be combined into a single step
    - Pass the input image through a filterbank
    - Filterbank setup : F filters of support $P = P_x \times P_y$ (patch spatial dimensions)
  - Collaborative filtering : Single convolution in the patch dimension after grouping
  - Adjoint NL operator : straightforward implementation as well



Non-local Regularization Operator

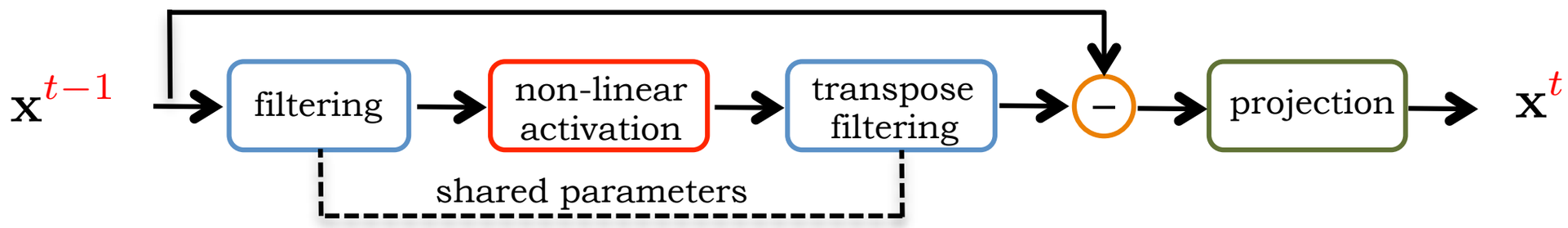
- Construction of the Non-Local operator
  - Patch extraction & patch-transform can be combined into a single step
    - Pass the input image through a filterbank
    - Filterbank setup : F filters of support $P = P_x \times P_y$ (patch spatial dimensions)
  - Collaborative filtering : Single convolution in the patch dimension after grouping
  - Adjoint NL operator : straightforward implementation as well



- Efficient NL implementation
  - Highly efficient specialized CPU & GPU libraries for convolutions
  - CUDNN (GPU), OMP (CPU multi-threading)

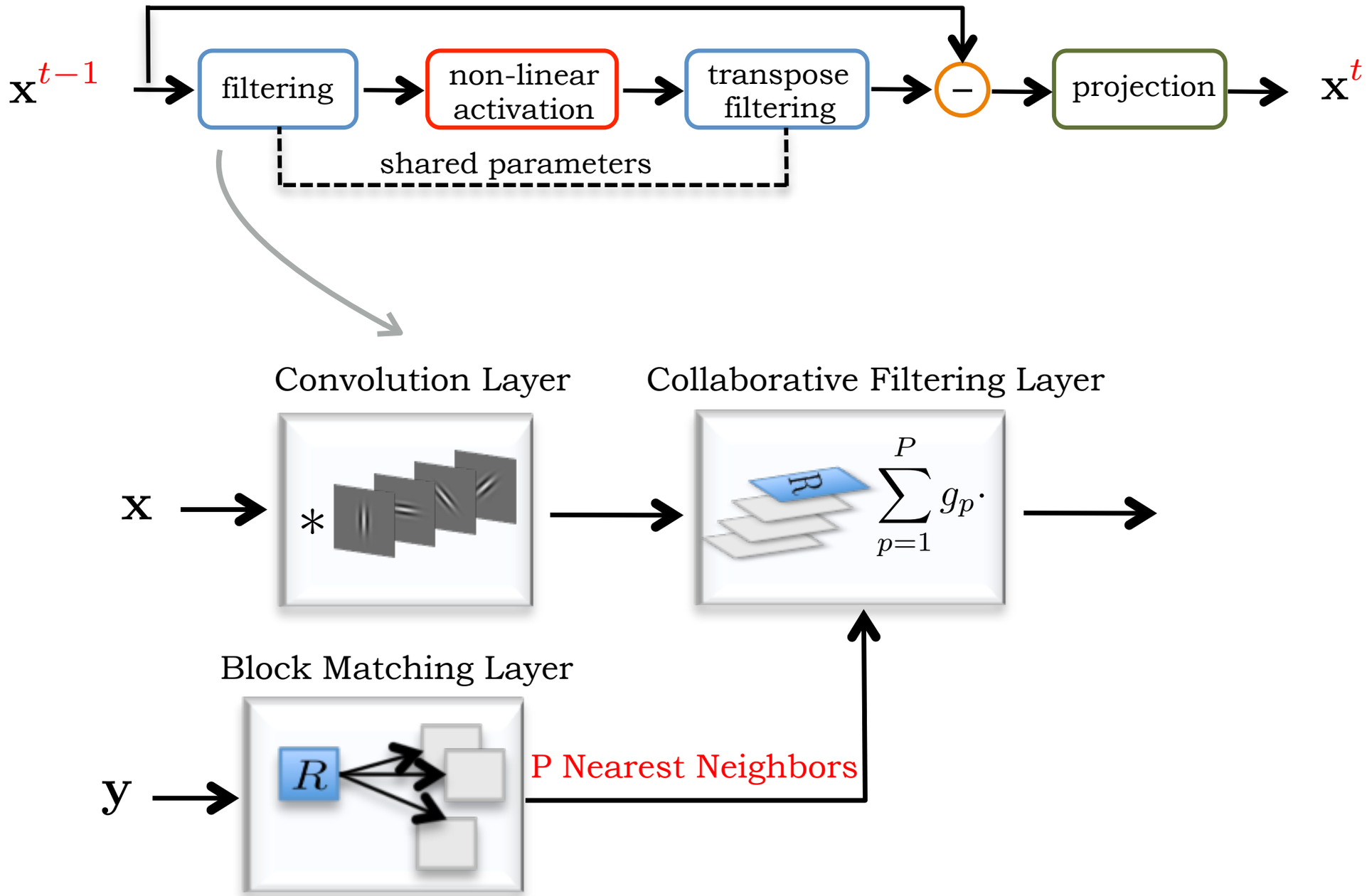
Non-local Network Architecture

- Cascade of composite layers (“stages”)



Non-local Network Architecture

- Cascade of composite layers (“stages”)



Discriminative Network Training

- Training of a S-stage network
  - Network parameters : $\boldsymbol{\Theta} = \left[\boldsymbol{\Theta}^1, \ldots, \boldsymbol{\Theta}^S\right], \boldsymbol{\Theta}^t = \left\{s^t, \mathbf{v}^t, \mathbf{g}^t, \boldsymbol{\pi}^t, \alpha^t\right\}$
- Learning using a loss-minimization strategy
  - Q pairs of training data $\left\{\mathbf{y}_{(q)}, \mathbf{x}_{(q)}\right\}_{q=1}^{Q}$
  - Different learned parameters for each stage to increase the capacity of the network
  - Loss function : negative PSNR
- Loss-minimization Strategies
  - Greedy approach - stage independent learning : $\mathcal{L}\left(\boldsymbol{\Theta}^t\right) = \sum_{q=1}^{Q} \ell\left(\hat{\mathbf{x}}_{(q)}^t, \mathbf{x}_{(q)}\right)$
  - Joint learning of all stages : $\mathcal{L}\left(\boldsymbol{\Theta}\right) = \sum_{q=1}^{Q} \ell\left(\hat{\mathbf{x}}_{(q)}^S, \mathbf{x}_{(q)}\right)$
- Minimization algorithm — ADAM
  - Variant of SGD that involves adaptive normalization of the learning rate
  - Derivative computation with back-propagation

Trained Models for Image Denoising

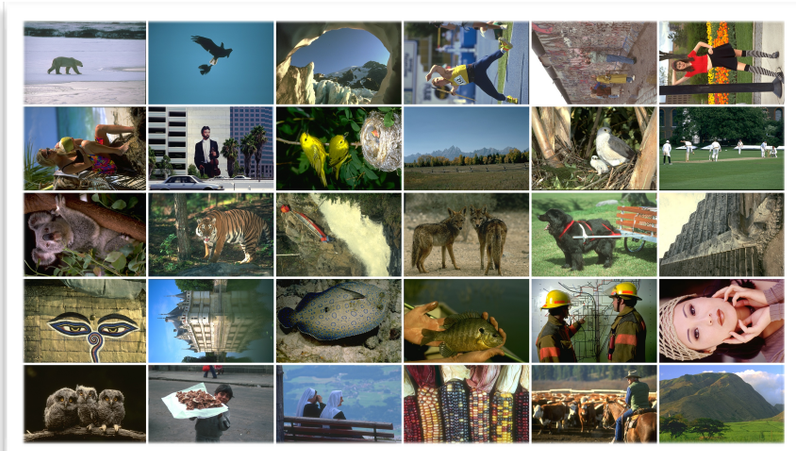
- Deep residual denoising networks for color & grayscale images
  - Grayscale and color local / non-local nets (5-stage feedforward networks)
  - A single network can handle a wide range of noise levels (σ = 5 … 55)
  - In the non-local deep nets similar patches are searched in a window 31 x 31
  - Group of non-local operator with K=8 closest neighbors

- Training data
  - Berkeley database (500 color / grayscale images)
  - For training : 400 images of size 180 x 180
  - For validation : 100 images of size 180 x 180

- Network training
  - Network initialization : greedy training with ADAM (100 epochs per stage)
  - Final network parameters : joint training with ADAM (100 epochs)
  - Training on NVIDIA GeForce 1080 Ti GPU

Comparisons

- Restoration performance evaluation on 68 BSDS images of size 321 x 481
- Comparisons against **state-of-the-art** methods
  - Three patch-based methods : BM3D [Dabov et al. `07], EPLL [Zoran & Weiss `11], WNNM [Gu et al. `14]
  - ML method : DCNN [Zhang et al. `17] : Deep network with approximately **14x** (**7x** for color images) more parameters and **2x** more deep
- Grayscale / color results



Methods	Noise level - σ (std.) 5	10	15	20	25	30	35	40	45	50	55	avg.
BM3D [8]	37.57	33.30	31.06	29.60	28.55	27.74	27.07	26.45	25.99	25.60	25.26	28.93
EPLL [45]	37.55	33.36	31.18	29.73	28.67	27.84	27.16	26.58	26.09	25.71	25.34	29.02
WNMM [15]	37.76	33.55	31.31	29.83	28.73	27.94	27.28	26.72	26.26	25.85	25.49	29.16
DCNN [42]	37.68	33.72	31.60	30.19	29.15	28.33	27.66	27.10	26.62	26.21	25.80	29.46
UNet_5	37.59	33.54	31.38	29.93	28.84	28.01	27.38	26.85	26.38	25.95	25.53	29.22
UNLNet_5	37.62	33.62	31.47	30.04	28.96	28.13	27.50	26.96	26.48	26.04	25.64	29.32
$\text{UNLNet}_5^{\text{orc}}$	37.79	33.97	31.95	30.59	29.51	28.54	27.97	27.47	26.97	26.41	25.80	29.72

Noise σ (std.)	Methods CBM3D [8]	CDCNN [42]	CUNet_5	CUNLNet_5	$\text{CUNLNet}_5^{\text{orc}}$
5	40.24	40.11	40.31	40.39	40.54
10	35.88	36.11	36.08	36.20	36.70
15	33.49	33.88	33.78	33.90	34.58
20	31.88	32.36	32.21	32.34	33.11
25	30.68	31.22	31.03	31.17	31.95
30	29.71	30.31	30.06	30.24	31.06
35	28.86	29.57	29.37	29.53	30.37
40	28.06	28.94	28.77	28.91	29.75
45	27.82	28.39	28.23	28.37	29.19
50	27.36	27.91	27.74	27.89	28.65
55	26.95	27.45	27.27	27.44	28.10
avg.	30.99	31.48	31.35	31.49	32.18

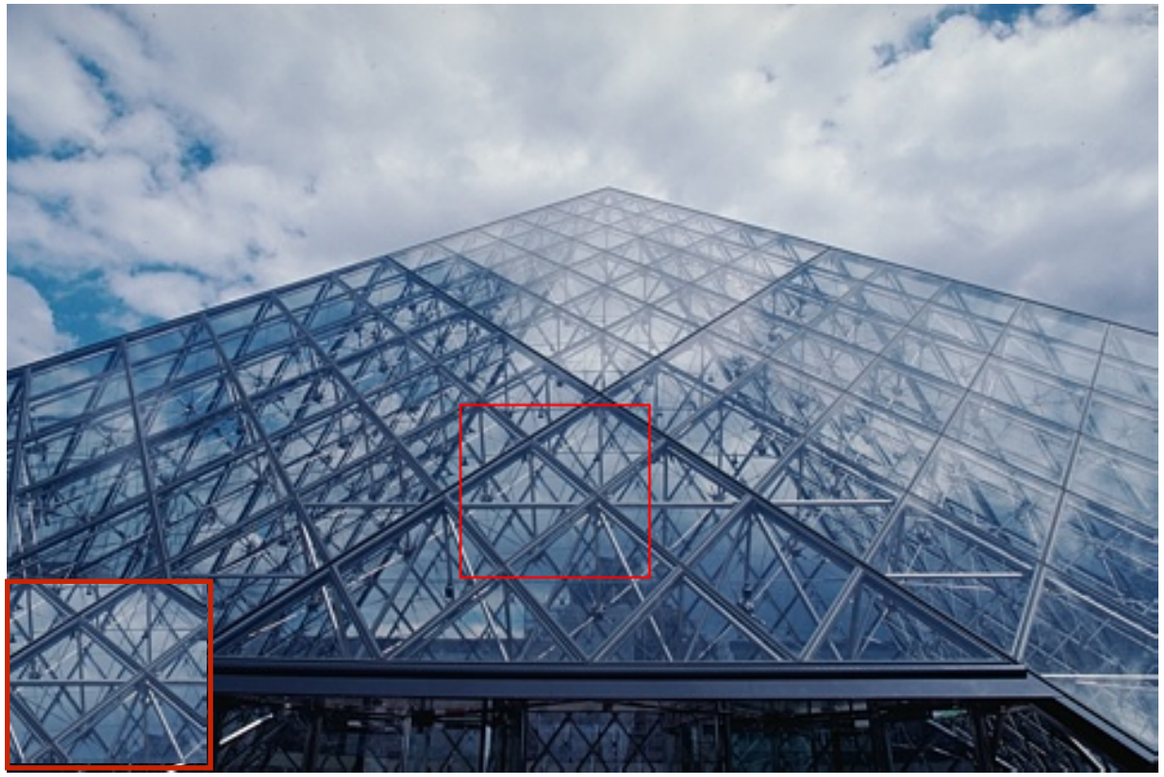
Grayscale Image Denoising

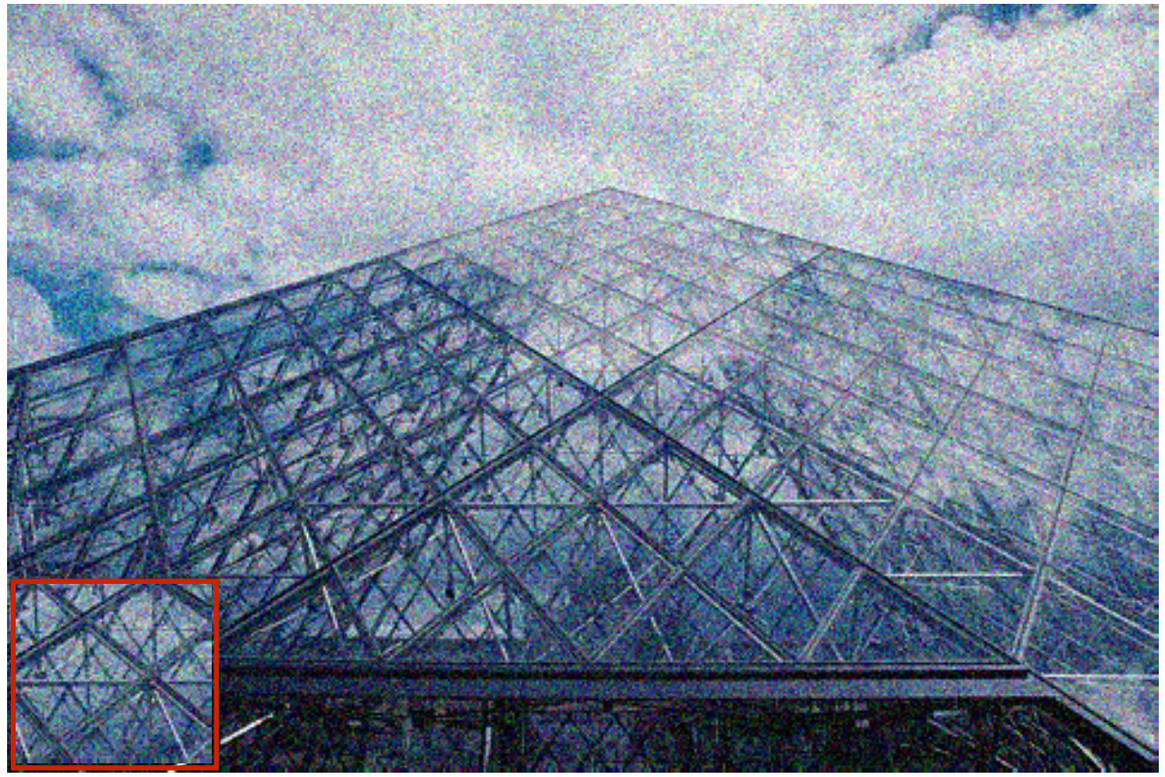
Noisy (std =20 ; PSNR = 22.10 dB)

EPLL (PSNR = 31.54 dB)

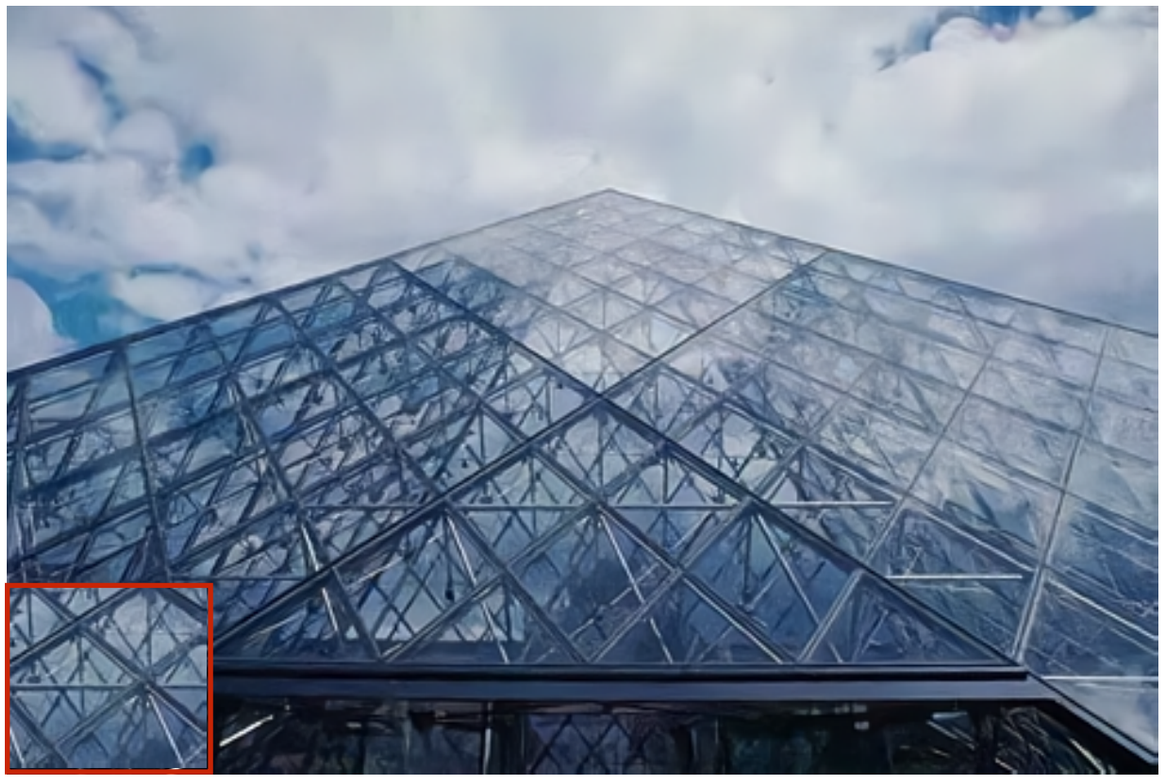
UNet_5 (PSNR = 31.71 dB)

Color Image Denoising

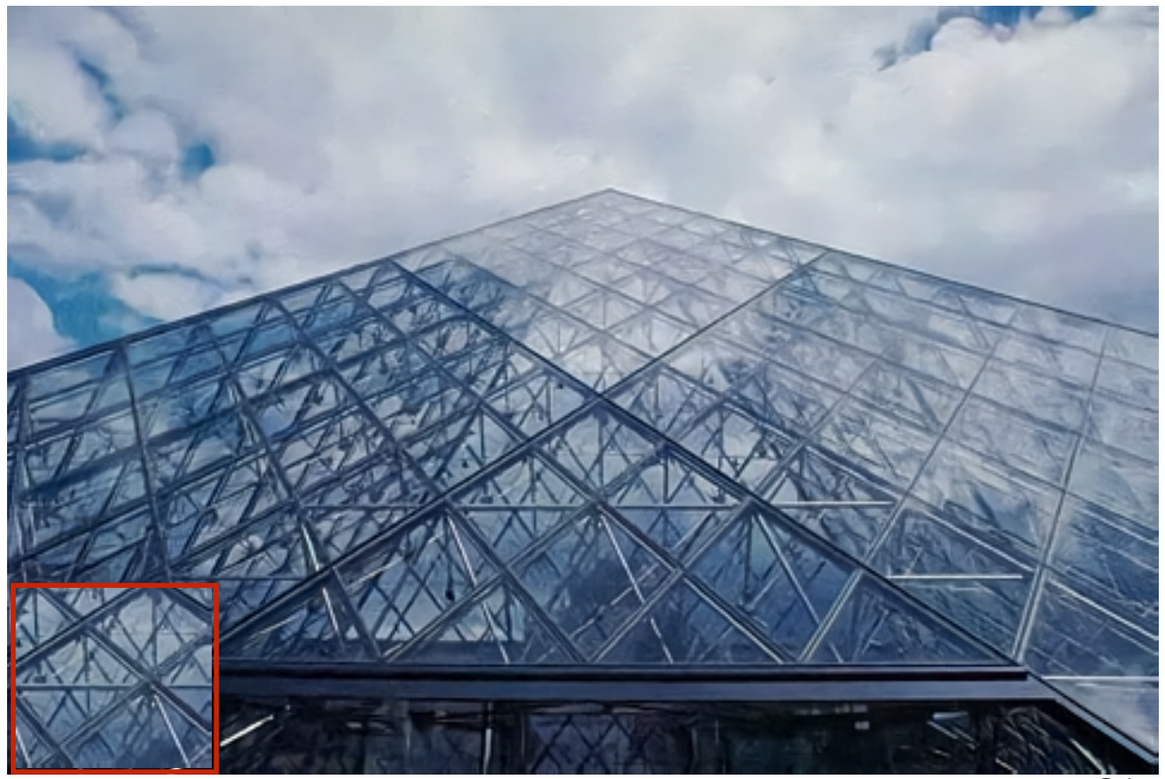




Noisy (std = 30 ; PSNR = 18.57 dB)



DCNN (PSNR = 29.08 dB)



CUNLNet_5 (PSNR = 29.13 dB)

Grayscale Image Denoising (real noise)

Noisy

BM3D

DCNN

UNet_5

Color Image Denoising (real noise)

Noisy

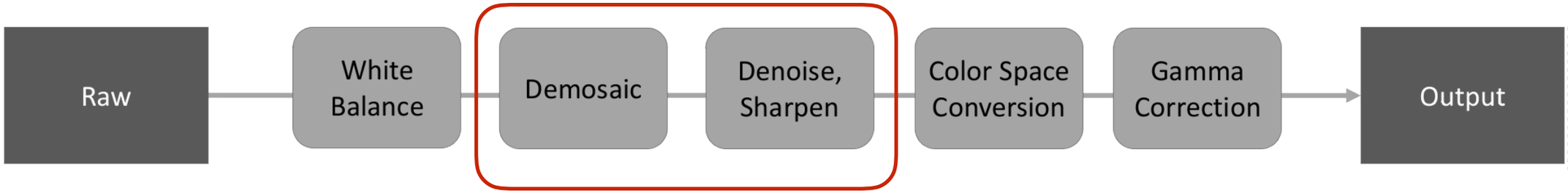
CBM3D

Noise Clinic

CUNLNet_5

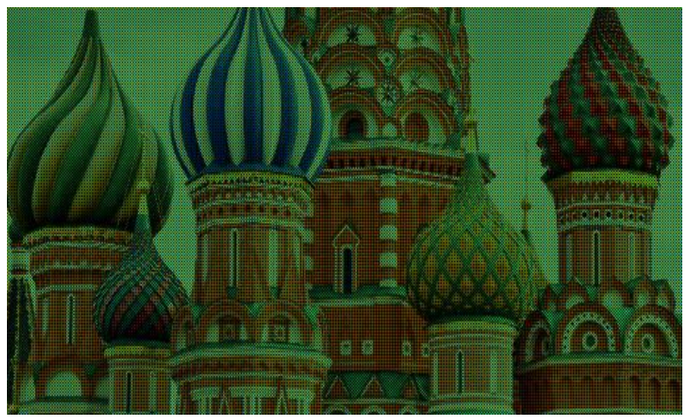
Image Demosaicking - Denoising

- First steps in the camera ISP :
  - Conversion of intensity readings to color images



- **Challenging** problem
  - Only one third of the intensities are known
  - Measurements are further distorted by noise

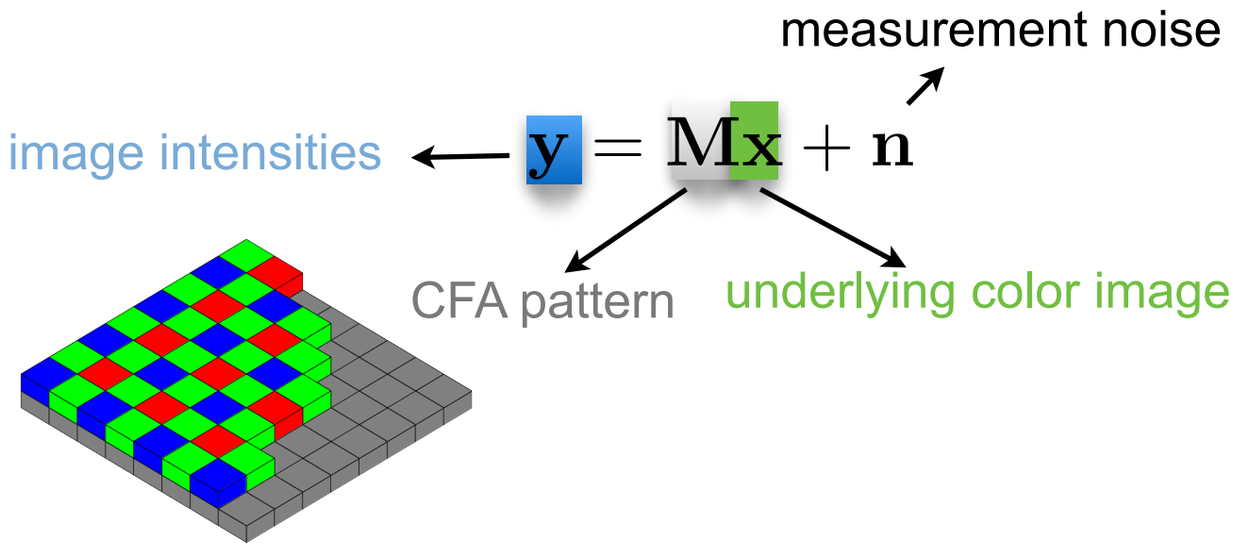
Noisy Light Intensity Measurements



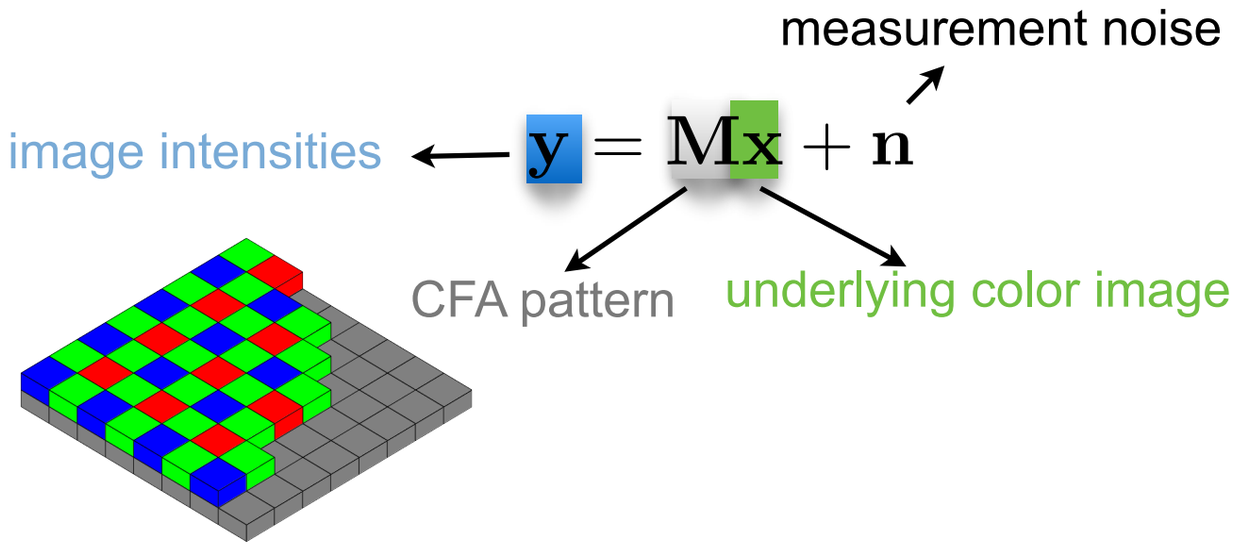
Demosaick & Denoise

End Result

Problem Formulation



Problem Formulation



- Solution of a minimization problem

$$\mathbf{x}^{\star} = \arg\min_{\mathbf{x}} \underbrace{\frac{1}{2\sigma^2} \|\mathbf{y} - \mathbf{M}\mathbf{x}\|_2^2 + r(\mathbf{x})}_{Q(\mathbf{x})}$$

- Challenges to deal with :
  - coupling of the CFA pattern with the solution
  - non-trivial way to minimize the objective function
  - choice of an appropriate regularizer

Majorization-Minimization framework

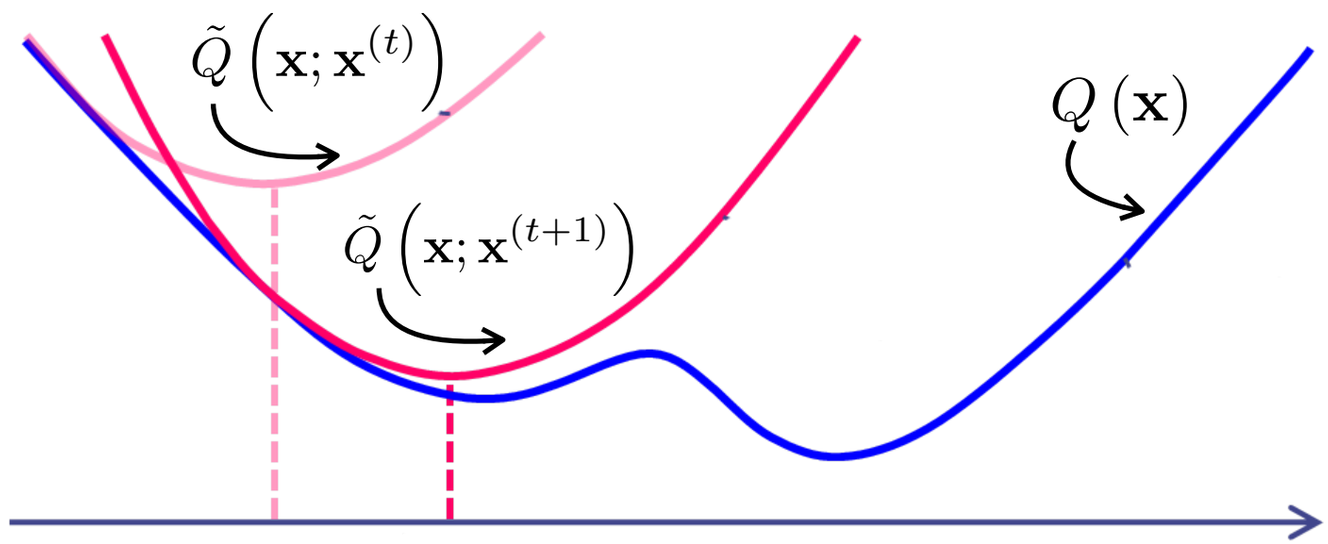
- Iterative minimization approach of the form :

$$\mathbf{x}^{(t+1)} = \arg\min_{\mathbf{x}} \tilde{Q}\left(\mathbf{x}; \mathbf{x}^{(t)}\right)$$

- $\tilde{Q}\left(\mathbf{x}; \mathbf{x}^{(t)}\right)$ is a majorizer of the objective function that satisfies the conditions:

$$(\alpha) \quad \tilde{Q}\left(\mathbf{x}; \mathbf{x}^{(t)}\right) > Q\left(\mathbf{x}\right), \forall \mathbf{x} \neq \mathbf{x}^{(t)}$$

$$(\beta) \quad \tilde{Q}\left(\mathbf{x}^{(t)}; \mathbf{x}^{(t)}\right) = Q\left(\mathbf{x}^{(t)}\right)$$



MM for Demosaicking-Denoising

- Derive a majorizer by majoring the data fidelity term

$$\tilde{d}\left(\mathbf{x};\mathbf{x}^{(t)}\right) = \underbrace{\frac{1}{2\sigma^2}\left\|\mathbf{y}-\mathbf{M}\mathbf{x}\right\|_2^2}_{d(\mathbf{x})} + g\left(\mathbf{x},\mathbf{x}^{(t)}\right)$$

$$g\left(\mathbf{x},\mathbf{x}^{(t)}\right) = \frac{1}{2\sigma^2}\left(\mathbf{x}-\mathbf{x}^{(t)}\right)^T\left[\alpha\mathbf{I}-\mathbf{M}^T\mathbf{M}\right]\left(\mathbf{x}-\mathbf{x}^{(t)}\right)$$

- $d\left(\cdot,\cdot\right)$ is a valid majorizer iff $g\left(\mathbf{x},\mathbf{y}\right) > 0\,, \forall \mathbf{x} \neq \mathbf{y}$ and $g\left(\mathbf{x},\mathbf{x}\right) = 0$
  - $\alpha\mathbf{I}-\mathbf{M}^T\mathbf{M} = \alpha\mathbf{I}-\mathbf{M}$ is positive definite $\Rightarrow \;\; \alpha > \left\|\mathbf{M}\right\| = 1$

MM for Demosaicking-Denoising

- Derive a majorizer by majoring the data fidelity term

$$\tilde{d}\left(\mathbf{x};\mathbf{x}^{(t)}\right)=\underbrace{\frac{1}{2\sigma^2}\left\|\mathbf{y}-\mathbf{M}\mathbf{x}\right\|_2^2}_{d(\mathbf{x})}+g\left(\mathbf{x},\mathbf{x}^{(t)}\right)$$

$$g\left(\mathbf{x},\mathbf{x}^{(t)}\right)=\frac{1}{2\sigma^2}\left(\mathbf{x}-\mathbf{x}^{(t)}\right)^T\left[\alpha\mathbf{I}-\mathbf{M}^T\mathbf{M}\right]\left(\mathbf{x}-\mathbf{x}^{(t)}\right)$$

- $d\left(\cdot,\cdot\right)$ is a valid majorizer iff $g\left(\mathbf{x},\mathbf{y}\right)>0\,,\forall\mathbf{x}\neq\mathbf{y}$ and $g\left(\mathbf{x},\mathbf{x}\right)=0$
  - $\alpha\mathbf{I}-\mathbf{M}^T\mathbf{M}=\alpha\mathbf{I}-\mathbf{M}$ is positive definite $\Rightarrow\ \alpha>\left\|\mathbf{M}\right\|=1$
- Overall majorizer is of the form :

$$\begin{aligned}\tilde{Q}\left(\mathbf{x};\mathbf{x}^{(t)}\right)&=\tilde{d}\left(\mathbf{x};\mathbf{x}^{(t)}\right)+r\left(\mathbf{x}\right)\\&=\frac{1}{2\left(\sigma\sqrt{\alpha}\right)^2}\left\|\mathbf{x}-\mathbf{z}\right\|_2^2+r\left(\mathbf{x}\right)+\text{const.}\end{aligned}$$

$$\mathbf{z}=\mathbf{y}+(\mathbf{I}-\mathbf{M})\mathbf{x}^{(t)}$$

MM for Demosaicking-Denoising

- Derive a majorizer by majoring the data fidelity term

$$\tilde{d}\left(\mathbf{x};\mathbf{x}^{(t)}\right)=\underbrace{\frac{1}{2\sigma^2}\left\|\mathbf{y}-\mathbf{M}\mathbf{x}\right\|_2^2}_{d(\mathbf{x})}+g\left(\mathbf{x},\mathbf{x}^{(t)}\right)$$

$$g\left(\mathbf{x},\mathbf{x}^{(t)}\right)=\frac{1}{2\sigma^2}\left(\mathbf{x}-\mathbf{x}^{(t)}\right)^T\left[\alpha\mathbf{I}-\mathbf{M}^T\mathbf{M}\right]\left(\mathbf{x}-\mathbf{x}^{(t)}\right)$$

- $d\left(\cdot,\cdot\right)$ is a valid majorizer iff $g\left(\mathbf{x},\mathbf{y}\right)>0\,,\forall\mathbf{x}\neq\mathbf{y}$ and $g\left(\mathbf{x},\mathbf{x}\right)=0$
  - $\alpha\mathbf{I}-\mathbf{M}^T\mathbf{M}=\alpha\mathbf{I}-\mathbf{M}$ is positive definite $\Rightarrow\ \alpha>\left\|\mathbf{M}\right\|=1$
- Overall majorizer is of the form :

$$\tilde{Q}\left(\mathbf{x};\mathbf{x}^{(t)}\right)=\tilde{d}\left(\mathbf{x};\mathbf{x}^{(t)}\right)+r\left(\mathbf{x}\right)$$

denoising objective function

$$=\frac{1}{2\left(\sigma\sqrt{\alpha}\right)^2}\left\|\mathbf{x}-\mathbf{z}\right\|_2^2+r\left(\mathbf{x}\right)+\text{const.}$$

$$\mathbf{z}=\mathbf{y}+(\mathbf{I}-\mathbf{M})\mathbf{x}^{(t)}$$

Residual Denoising Network

- Recast the problem of image demosaicking to a series of **image denoising problems**

$$\tilde{Q}\left(\mathbf{x};\mathbf{x}^{(t)}\right)=\frac{1}{2\left(\sigma\sqrt{\alpha}\right)^{2}}\left\|\mathbf{x}-\mathbf{z}\right\|_{2}^{2}+r\left(\mathbf{x}\right)+\text{const.}$$

- In each iteration the estimate of the demosaicked image is refined

Residual Denoising Network

- Recast the problem of image demosaicking to a series of **image denoising problems**

$$\tilde{Q}\left(\mathbf{x};\mathbf{x}^{(t)}\right)=\frac{1}{2\left(\sigma\sqrt{\alpha}\right)^{2}}\left\|\mathbf{x}-\mathbf{z}\right\|_{2}^{2}+r\left(\mathbf{x}\right)+\text{const.}$$

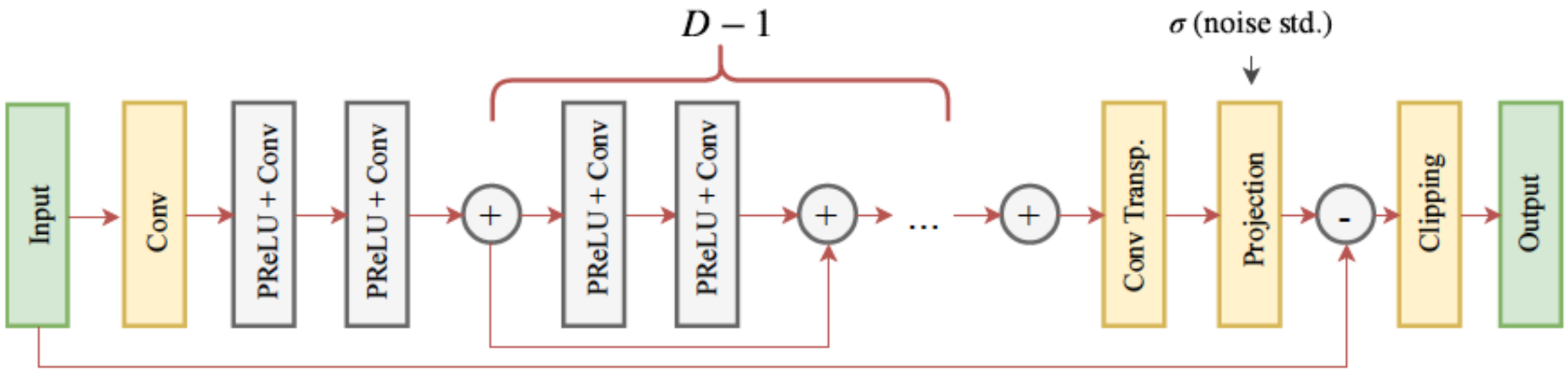
- In each iteration the estimate of the demosaicked image is refined
- Instead of solving an optimization problem we employ a denoising network

Residual Denoising Network

- Recast the problem of image demosaicking to a series of **image denoising problems**

$$\tilde{Q}\left(\mathbf{x};\mathbf{x}^{(t)}\right)=\frac{1}{2\left(\sigma\sqrt{\alpha}\right)^{2}}\left\|\mathbf{x}-\mathbf{z}\right\|_{2}^{2}+r\left(\mathbf{x}\right)+\text{const.}$$

- In each iteration the estimate of the demosaicked image is refined
- Instead of solving an optimization problem we employ a denoising network



Iterative Joint Demosaicking-Denoising

Algorithm 1: The proposed demosaicking network described as an iterative process. The ResDNet parameters are shared across all iterations.

Input: M: CFA, y: input, K: iterations, $w \in \mathbb{R}^K$: extrapolation weights, σ: estimated noise, $\gamma \in \mathbb{R}^K$: projection parameters

$\mathbf{x}^{(0)} = \mathbf{0}$;

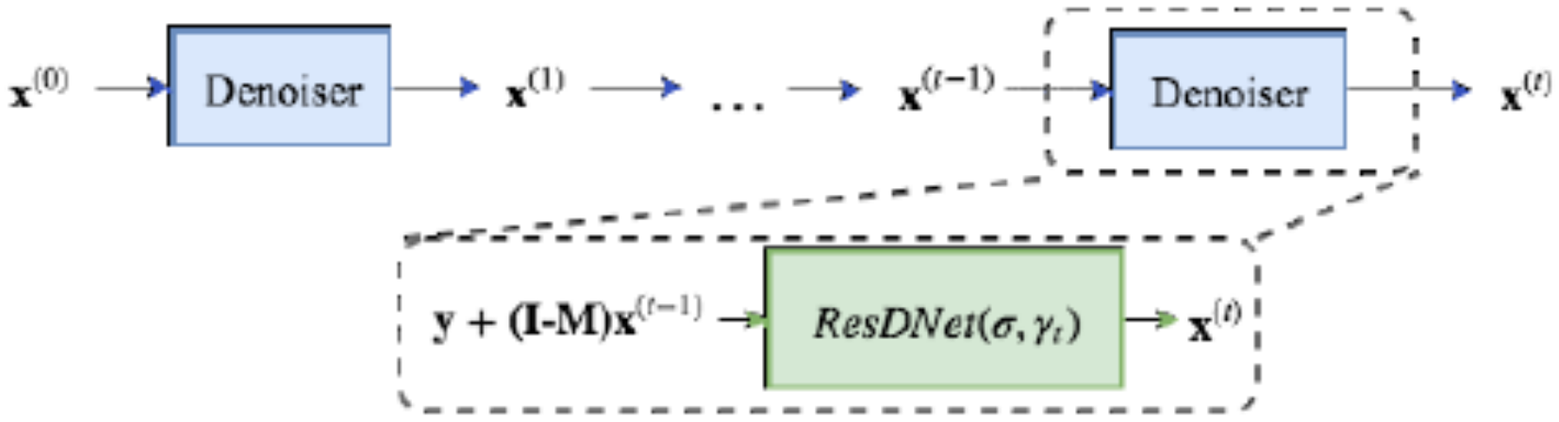
Initialize $\mathbf{x}^{(1)}$ using y;

for $i \leftarrow 1$ **to** K **do**

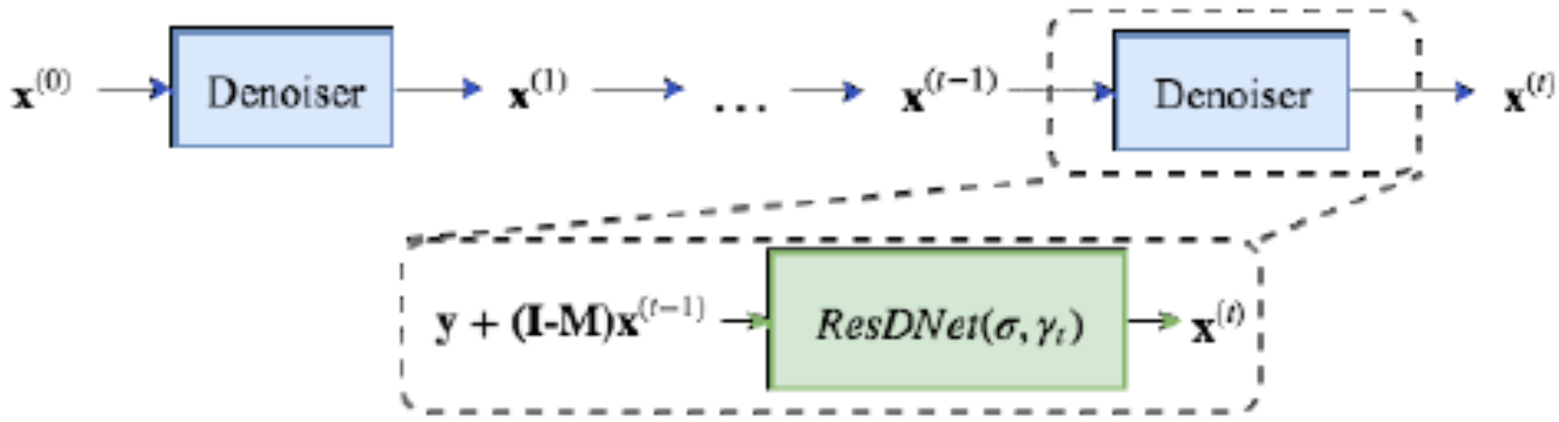
  $\mathbf{u} = \mathbf{x}^{(i)} + w_i(\mathbf{x}^{(i)} - \mathbf{x}^{(i-1)})$;

  $\mathbf{x}^{(i+1)} = \text{ResDNet}((\mathbf{I} - \mathbf{M})\mathbf{u} + \mathbf{y}, \sigma, \gamma_i)$;

end

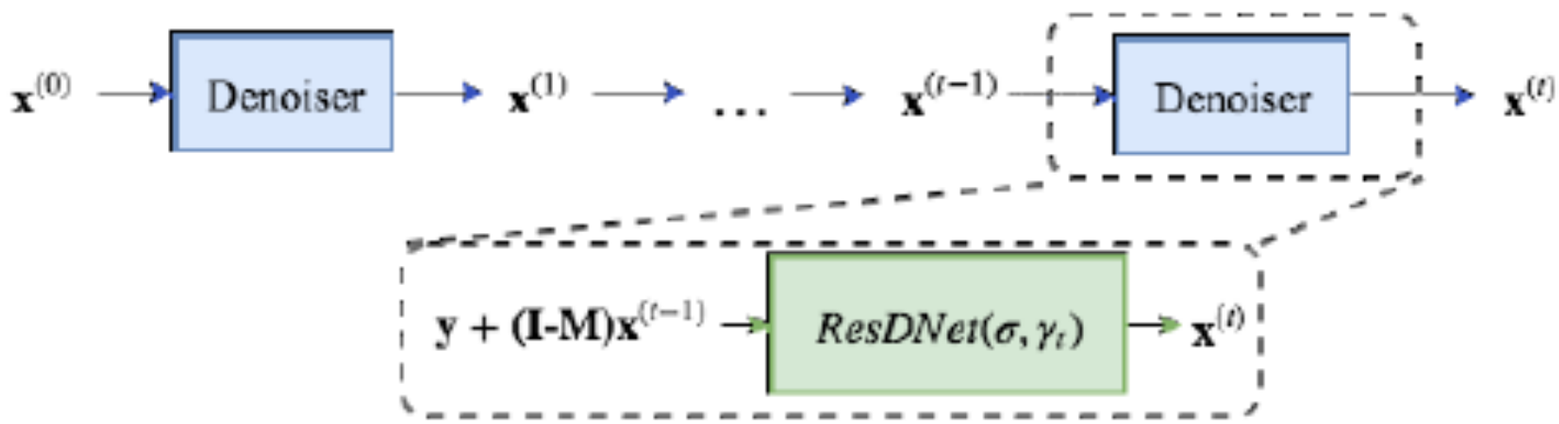


Iterative Joint Demosaicking-Denoising



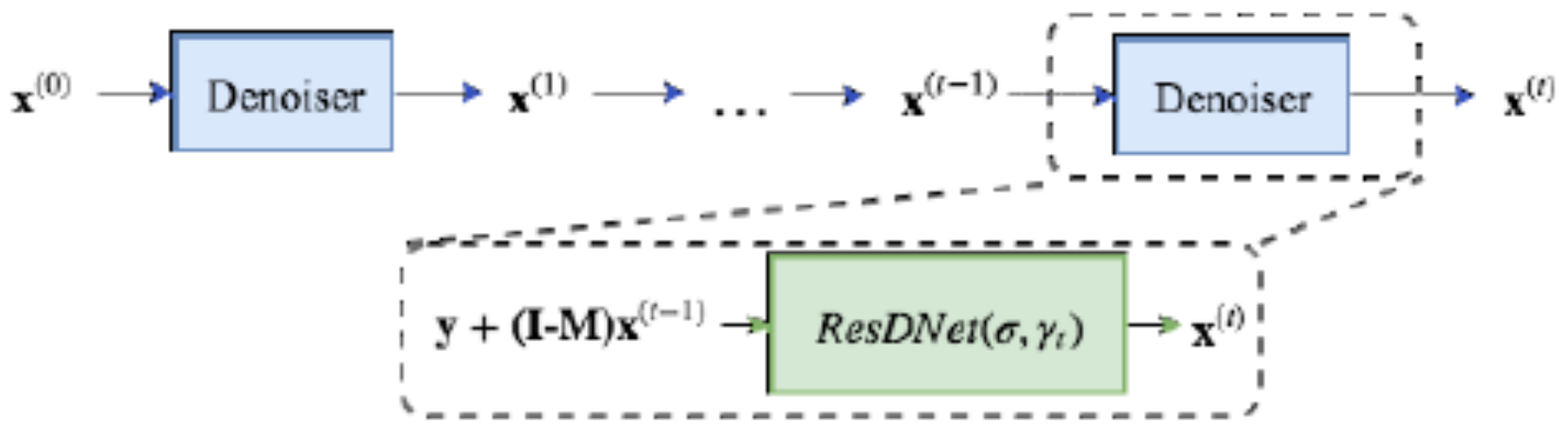
- The network parameters are **shared** across iterations
  - ✓ small memory footprint of the network

Iterative Joint Demosaicking-Denoising

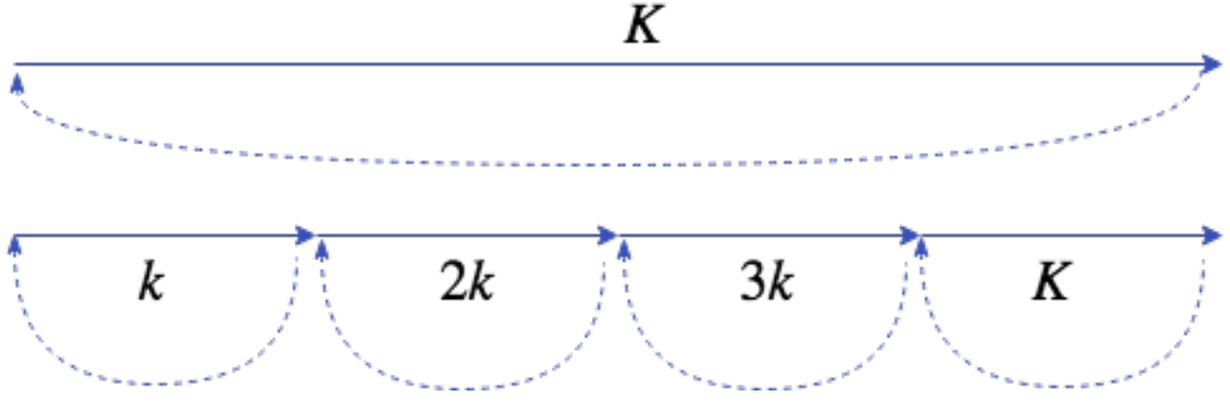


- The network parameters are **shared** across iterations
  - ✓ small memory footprint of the network
- Challenge: How to efficiently train the network by back-propagation?
  - Back propagation through time (BPTT) is **extremely memory demanding**

Iterative Joint Demosaicking-Denoising



- The network parameters are **shared** across iterations
  - ✓ small memory footprint of the network
- Challenge: How to efficiently train the network by back-propagation?
  - Back propagation through time (BPTT) is **extremely memory demanding**
- Use instead Truncated BPTT
  - adopted approach in natural language processing (NLP)
  - first to use in image restoration or computer vision problems
  - ✓ We can train arbitrarily deep demosaicking networks



Comparisons

Noise-free scenario on synthetic data

	Kodak	McM	VDP	Moire
Bilinear	32.9	32.5	25.2	27.6
Zhang (NLM) [2]	37.9	36.3	30.1	31.9
Hirakawa [49]	36.5	33.9	30.0	32.1
Getreuer [50]	38.1	36.1	30.8	32.5
Heide [5]	40.0	38.6	27.1	34.9
Klatzer [22]	35.3	30.8	28.0	30.3
Gharbi [15]	41.2	39.5	34.3	37.0
Kokkinos [23]	41.5	**39.7**	**34.5**	37.0
Tan [7]	**42.0**	39.0	-	-
Henz [16]	41.9	39.5	34.3	36.3
$MMNet_{10}$ (ours)	**42.0**	**39.7**	**34.5**	**37.1**

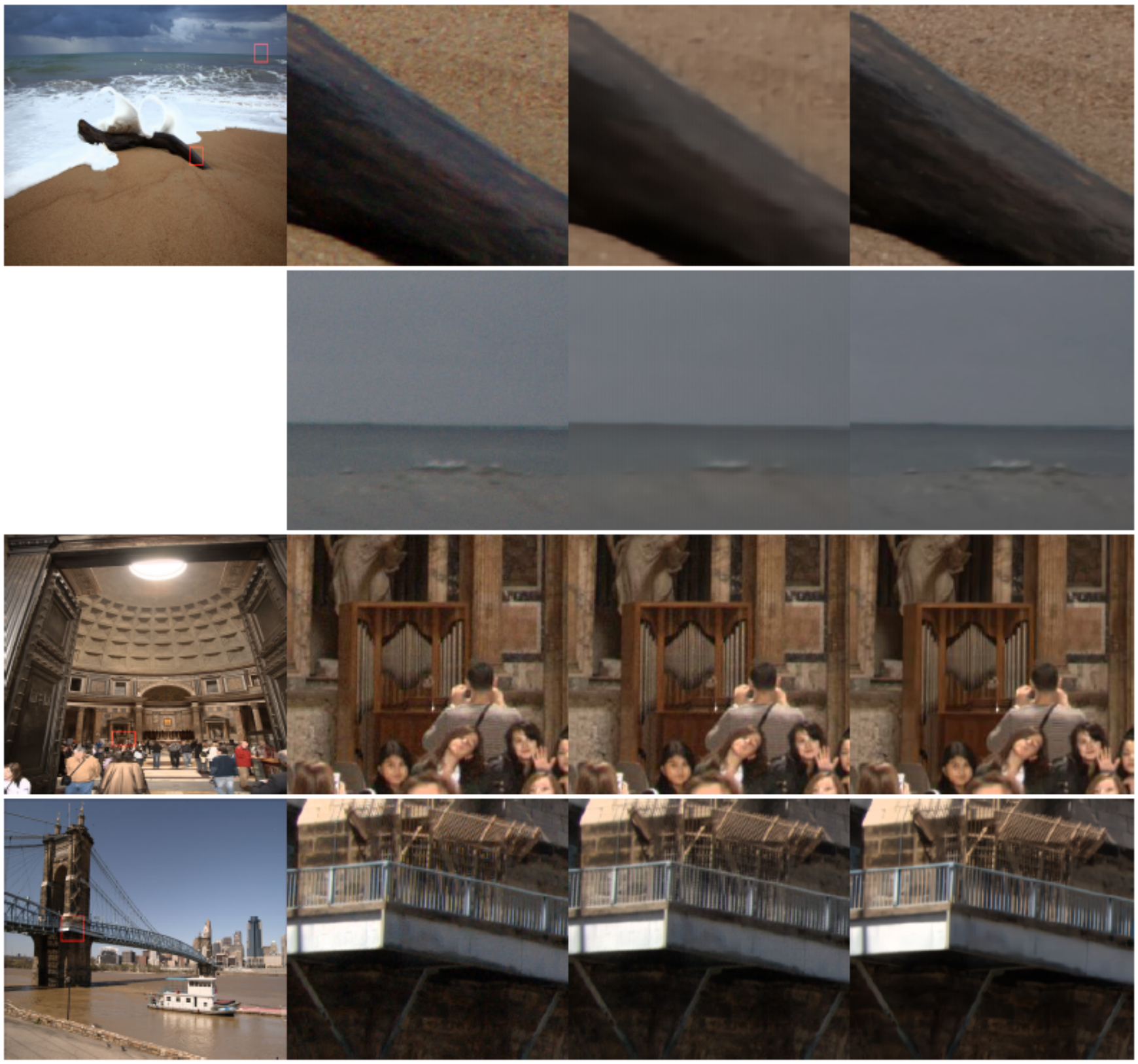
real-world data (MSR dataset) / Bayer CFA

	noisy		noise-free	
	linRGB	sRGB	linRGB	sRGB
Bilinear	-	-	30.9	24.9
Zhang(NLM) [2]	-	-	38.4	32.1
Hirakawa [50]	-	-	37.2	31.3
Heide [5]	-	-	40.0	33.8
Khasabi [11]	37.8	31.5	39.4	32.6
Klatzer [23]	38.8	32.6	40.9	34.6
Gharbi [15]	38.6	32.6	42.7	35.9
Kokkinos [24]	39.2	33.3	41.0	34.6
$MMNet_2$ (ours)	39.2	33.3	42.1	35.6
$MMNet_{20}$ (ours)	**40.1**	**34.2**	**42.8**	**36.4**

real-world data (MSR dataset) / Fuji XTrans CFA

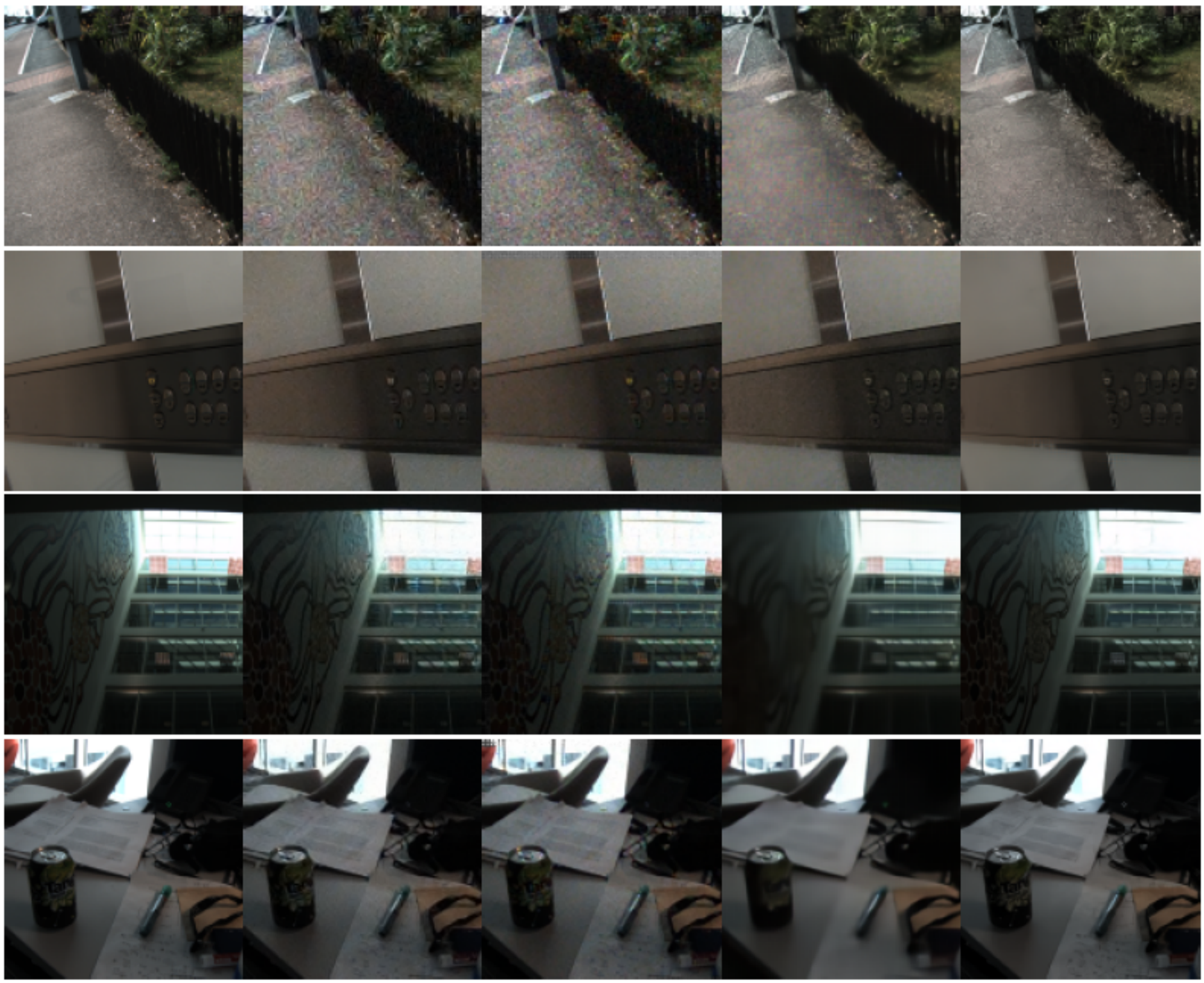
	noise-free	
	linear	sRGB
Trained on MSR Dataset:		
Khashabi [11]	36.9	30.6
Klatzer [23]	39.6	33.1
Gharbi [15]	39.7	33.2
Kokkinos [24]	39.9	33.7
$MMNet_8$ (ours)	40.2	34.0
$MMNet_{20}$ (ours)	**40.6**	**34.4**

Demosaicking of real raw data



Ours (full) | Hirakawa [50] (dcraw) | Gharbi et al. [15] | ours

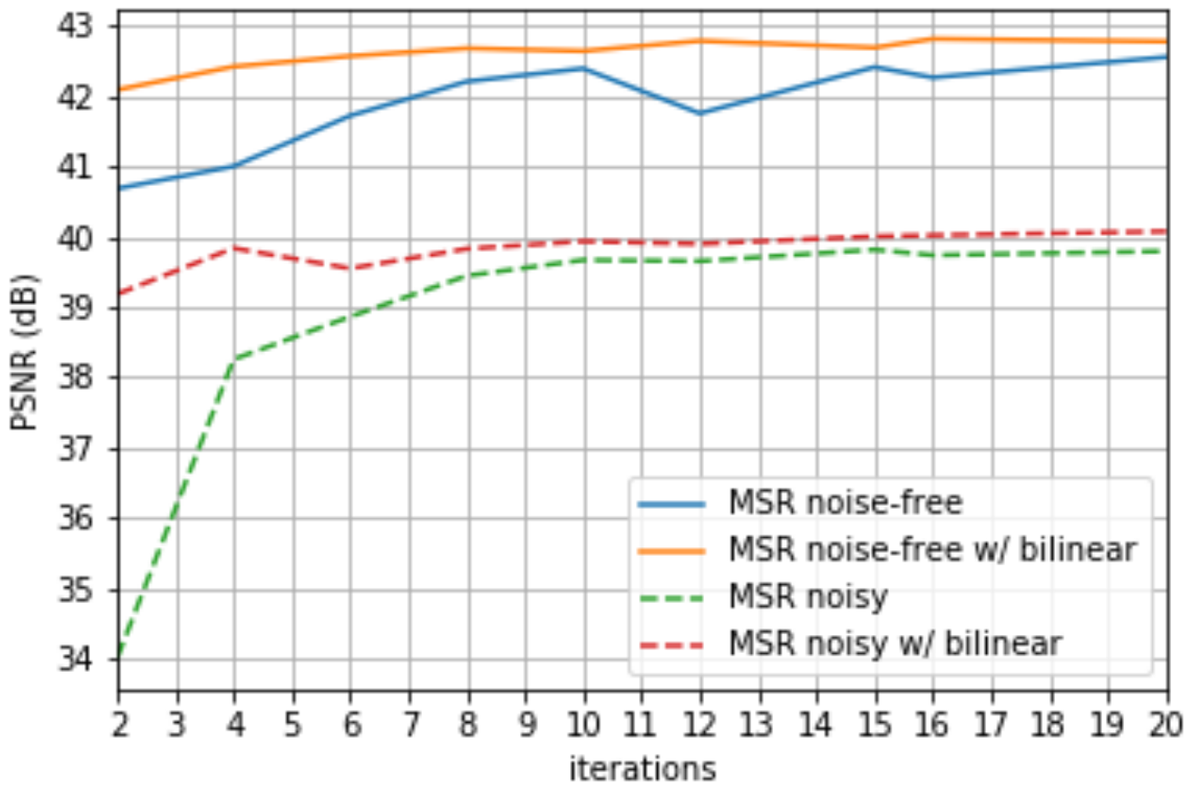
Demosaicking of MSR raw data

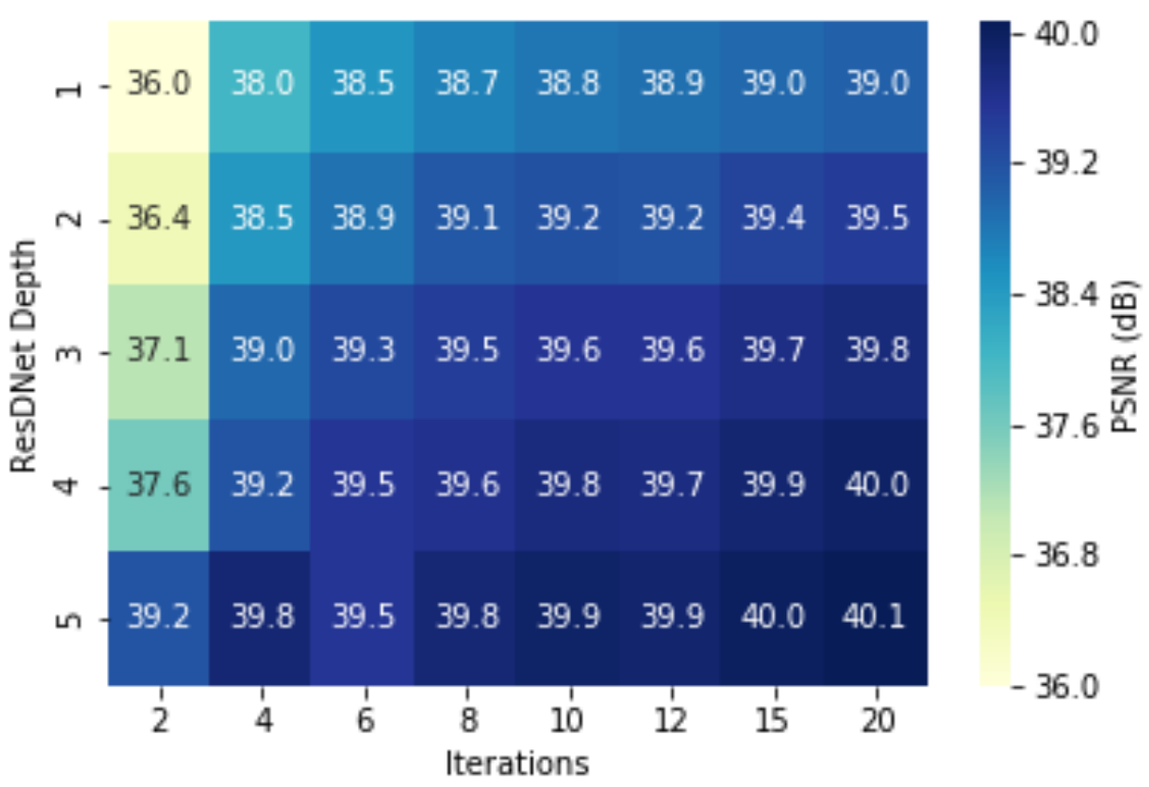


Reference | Hirakawa (dcraw)[50] | Zhang(NLM)[2] | Gharbi et al.[15] | ours

Ablation Study

- Our iterative approach provides a balance between the number of network parameters and its depth (iterations). Specifically we are able :
  - ✓ to retrieve the same performance when reducing the number of parameters by increasing the number of iterations
  - ✓ to be robust to the initialization by using a deep network architecture





Summary and Future Research Directions

- **Deep Learning for inverse imaging problems**
  - Inspiration from variational methods to design a deep network
  - Data-driven approach for learning the potential function and regularization operator
  - Non-local deep models : exploit the self-similarity property of natural images
  - A single set of learned parameters that handles a wide range of noise levels
  - Computational efficiency and increased robustness to degradation factors
  - Good generalization ability of the developed networks to real noise
  - **State-of-the-art** image restoration results using a considerably smaller network than the previous best-performing network
  - The proposed networks can serve as sub-solvers for dealing with other general inverse problems (**state-of-the-art** results in image **demosaicking**)

Summary and Future Research Directions

- **Deep Learning for inverse imaging problems**
  - Inspiration from variational methods to design a deep network
  - Data-driven approach for learning the potential function and regularization operator
  - Non-local deep models : exploit the self-similarity property of natural images
  - A single set of learned parameters that handles a wide range of noise levels
  - Computational efficiency and increased robustness to degradation factors
  - Good generalization ability of the developed networks to real noise
  - **State-of-the-art** image restoration results using a considerably smaller network than the previous best-performing network
  - The proposed networks can serve as sub-solvers for dealing with other general inverse problems (**state-of-the-art** results in image **demosaicking**)

- **Broad spectrum of applications**
  - Computational photography : deblurring, superresolution, demosaicking, …
  - Medical imaging : (P)MRI reconstruction, computed tomography
  - Seismic imaging : seismic data recovery - phase retrieval
  - Computer vision : optical flow estimation, dense stereo reconstruction

Main References

- S. Lefkimmiatis, “Non-Local Color Image Denoising with Convolutional Neural Networks”, IEEE Int. Conference on Computer Vision and Pattern Recognition, 2017
- S. Lefkimmiatis, “Universal Denoising Networks : A Novel CNN Architecture for Image Denoising, IEEE Int. Conference on Computer Vision and Pattern Recognition, 2018
- F. Kokkinos and S. Lefkimmiatis, “Deep Image Demosaicking using a Cascade of Convolutional Residual Denoising Networks”, European Conference on Computer Vision, 2018
- F. Kokkinos and S. Lefkimmiatis, “Iterative Residual Network for Deep Joint Image Demosaicking and Denoising” , arXiv:1807.06403

Main References

- S. Lefkimmiatis, “Non-Local Color Image Denoising with Convolutional Neural Networks”, IEEE Int. Conference on Computer Vision and Pattern Recognition, 2017
- S. Lefkimmiatis, “Universal Denoising Networks : A Novel CNN Architecture for Image Denoising, IEEE Int. Conference on Computer Vision and Pattern Recognition, 2018
- F. Kokkinos and S. Lefkimmiatis, “Deep Image Demosaicking using a Cascade of Convolutional Residual Denoising Networks”, European Conference on Computer Vision, 2018
- F. Kokkinos and S. Lefkimmiatis, “Iterative Residual Network for Deep Joint Image Demosaicking and Denoising” , arXiv:1807.06403

software and publications available at: http://cig.skoltech.ru

Thank you for your attention

Non-local Regularization Operator

- Construction of the Non-local Operator
  - Patch Extraction : $\mathbf{x}_r = \mathbf{P}_r \mathbf{x}$

    $\mathbf{P}_r \in \mathbb{R}^{P \times N}$: Binary matrix

  - Nearest Neighbors Search : $i_r = \{i_{r,1}, i_{r,2}, \ldots, i_{r,K}\}$, with $r = 1, \ldots, R$

    closest neighbors indices
  - Patch Transform : $\mathbf{f}_r = \mathbf{F}\mathbf{x}_r, \quad \mathbf{F} \in \mathbb{R}^{F \times P}$
  - Grouping : $\mathbf{f}_{i_r} = \begin{bmatrix} \mathbf{f}_{i_{r,1}}^{\mathsf{T}} & \mathbf{f}_{i_{r,2}}^{\mathsf{T}} & \ldots & \mathbf{f}_{i_{r,K}}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^{F \cdot K}$
  - Collaborative filtering : $\mathbf{z}_r = \mathbf{W}\mathbf{f}_{i_r}, \ \mathbf{W} \in \mathbb{R}^{F \times (F \cdot K)} \qquad \left(\mathbf{z}_r = \sum_{k=1}^{K} w_k \mathbf{f}_{i_{r,k}} \right)$

Non-local Regularization Operator

- Construction of the Non-local Operator
  - Patch Extraction : $\mathbf{x}_r = \mathbf{P}_r \mathbf{x}$

    $\mathbf{P}_r \in \mathbb{R}^{P \times N}$: Binary matrix

  - Nearest Neighbors Search : $i_r = \{i_{r,1}, i_{r,2}, \ldots, i_{r,K}\}$, with $r = 1, \ldots, R$ — closest neighbors indices
  - Patch Transform : $\mathbf{f}_r = \mathbf{F}\mathbf{x}_r, \quad \mathbf{F} \in \mathbb{R}^{F \times P}$
  - Grouping : $\mathbf{f}_{i_r} = \begin{bmatrix} \mathbf{f}_{i_{r,1}}^{\mathsf{T}} & \mathbf{f}_{i_{r,2}}^{\mathsf{T}} & \ldots & \mathbf{f}_{i_{r,K}}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^{F \cdot K}$
  - Collaborative filtering : $\mathbf{z}_r = \mathbf{W}\mathbf{f}_{i_r}, \ \mathbf{W} \in \mathbb{R}^{F \times (F \cdot K)}$ $\left(\mathbf{z}_r = \sum_{k=1}^{K} w_k \mathbf{f}_{i_{r,k}} \right)$
- Non-local Operator - final form
  - Composition of linear operators : $\mathbf{L}_r \, \mathbf{x} = \left(\mathbf{W}\tilde{\mathbf{F}}\mathbf{P}_{i_r} \right) \mathbf{x}$ — block-diagonal matrix
  - Existence of the adjoint operator : $\mathbf{L}_r^{\mathsf{T}} = \mathbf{P}_{i_r}^{\mathsf{T}} \tilde{\mathbf{F}}^{\mathsf{T}} \mathbf{W}^{\mathsf{T}}$

Non-local Regularization Operator

- Construction of the Non-local Operator
  - Patch Extraction : $\mathbf{x}_r = \mathbf{P}_r \mathbf{x}$

    $\mathbf{P}_r \in \mathbb{R}^{P \times N}$: Binary matrix

  - Nearest Neighbors Search : $i_r = \{i_{r,1}, i_{r,2}, \ldots, i_{r,K}\}, \quad \text{with } r = 1, \ldots, R$

    closest neighbors indices
  - Patch Transform : $\mathbf{f}_r = \mathbf{F}\mathbf{x}_r, \quad \mathbf{F} \in \mathbb{R}^{F \times P}$
  - Grouping : $\mathbf{f}_{i_r} = \begin{bmatrix} \mathbf{f}_{i_{r,1}}^{\mathsf{T}} & \mathbf{f}_{i_{r,2}}^{\mathsf{T}} & \ldots & \mathbf{f}_{i_{r,K}}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^{F \cdot K}$
  - Collaborative filtering : $\mathbf{z}_r = \mathbf{W}\mathbf{f}_{i_r}, \quad \mathbf{W} \in \mathbb{R}^{F \times (F \cdot K)}$ $\left(\mathbf{z}_r = \sum_{k=1}^{K} w_k \mathbf{f}_{i_{r,k}} \right)$
- Non-local Operator - final form

  block-diagonal matrix
  - Composition of linear operators : $\mathbf{L}_r \, \mathbf{x} = \left(\mathbf{W}\tilde{\mathbf{F}}\mathbf{P}_{i_r} \right) \mathbf{x}$
  - Existence of the adjoint operator : $\mathbf{L}_r^{\mathsf{T}} = \mathbf{P}_{i_r}^{\mathsf{T}} \tilde{\mathbf{F}}^{\mathsf{T}} \mathbf{W}^{\mathsf{T}}$
- Machine-learning objective
  - Learn the patch transform & collaborative filtering coefficients from training data