HYPERSPECTRAL IMAGE FUSION USING FAST HIGH-DIMENSIONAL DENOISING

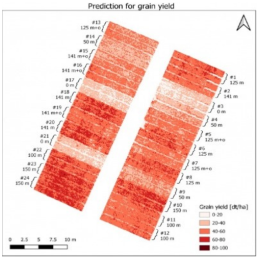
Pravin Nair, Unni V. S. and Kunal N. Chaudhury

Indian Institute of Science

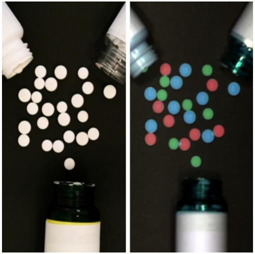
September 25, 2019

Hyperspectral imaging

(a) Environment Monitor



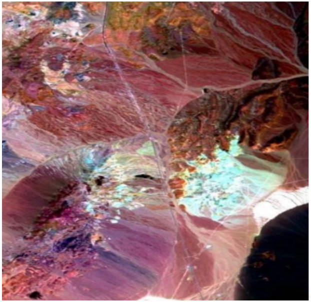
(b) Yield estimation

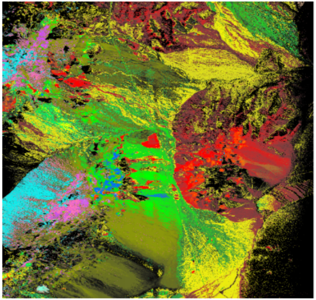


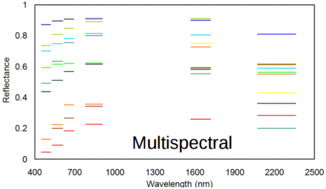
(c) Pharmaceuticals

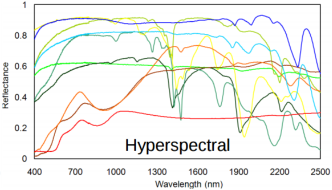
(d) Find finished goods

Spectral imaging





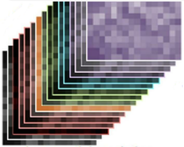




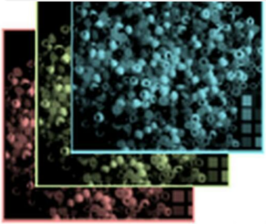
Hyperspectral remote sensing and it's potential, SAC.gov.in.

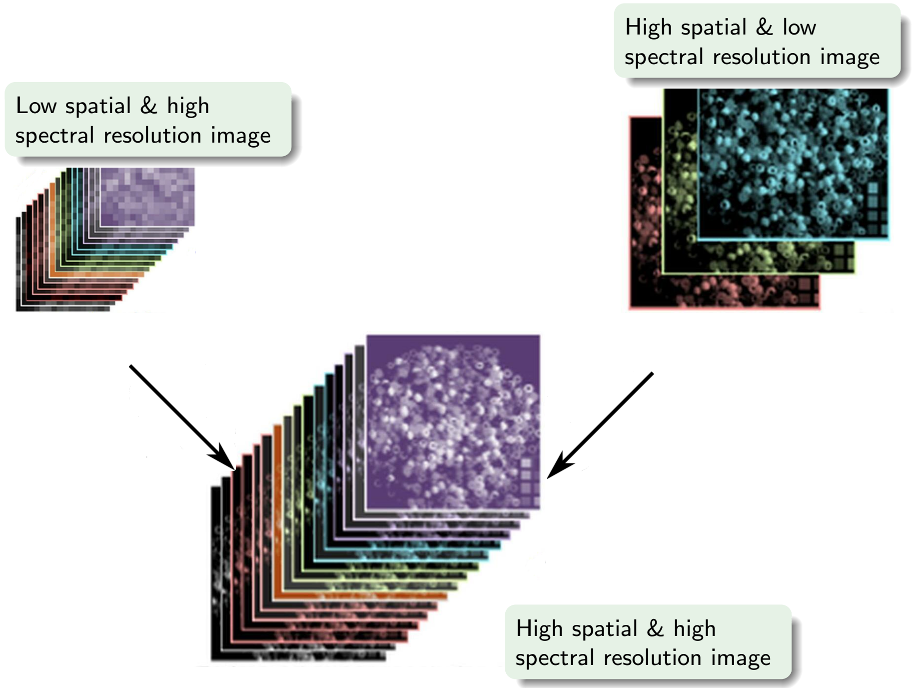
Hyperspectral image fusion

Low spatial & high spectral resolution image



High spatial & low spectral resolution image




Low spatial & high
spectral resolution image
High spatial & low
spectral resolution image
High spatial & high
spectral resolution image

Forward model

$$\mathbf{Y}_h = \mathbf{SBZ} + \mathbf{N}_h \quad \text{and} \quad \mathbf{Y}_m = \mathbf{ZR} + \mathbf{N}_m,$$

- $\mathbf{Z} \in \mathbb{R}^{n_m \times \ell_h}$ - Target image (reconstruction) with high spatial and spectral resolutions; it has ℓ_h bands and n_m pixels.
- $\mathbf{Y}_h \in \mathbb{R}^{n_h \times \ell_h}$ - Low resolution HS image, $n_h \ll n_m$.
- $\mathbf{Y}_m \in \mathbb{R}^{n_m \times \ell_m}$ - High resolution MS image, $\ell_m \ll \ell_h$.
- $\mathbf{B} \in \mathbb{R}^{n_m \times n_m}$ - Spatial blurring operator.
- $\mathbf{S} \in \mathbb{R}^{n_h \times n_m}$ - Sub-sampling (decimation) operator.
- $\mathbf{R} \in \mathbb{R}^{\ell_h \times \ell_m}$ - Spectral degradation operator.
- $\mathbf{N}_h$ and $\mathbf{N}_m$ - White Gaussian noise.

Yokoya et al., IEEE Geosci. Remote Sens. Mag, 2017.

Low-rank model

$$\mathbf{Y}_h = \mathbf{SBXE} + \mathbf{N}_h \quad \text{and} \quad \mathbf{Y}_m = \mathbf{XER} + \mathbf{N}_m.$$

Assumption

$$\mathbf{Z} \approx \mathbf{XE}$$

$\mathbf{E} \in \mathbb{R}^{\ell_s \times \ell_h}$ models a lower dimensional subspace ($\ell_s << \ell_h$).
$\mathbf{X} \in \mathbb{R}^{n_m \times \ell_s}$ is the projection of $\mathbf{Z}$ on the subspace.

Bioucas-Dias et al., IEEE Trans. Geosci. Remote Sens, 2008.

Low-rank model

$$\mathbf{Y}_h = \mathbf{SBXE} + \mathbf{N}_h \quad \text{and} \quad \mathbf{Y}_m = \mathbf{XER} + \mathbf{N}_m.$$

Assumption

$$\mathbf{Z} \approx \mathbf{XE}$$

$\mathbf{E} \in \mathbb{R}^{\ell_s \times \ell_h}$ models a lower dimensional subspace ($\ell_s << \ell_h$).
$\mathbf{X} \in \mathbb{R}^{n_m \times \ell_s}$ is the projection of $\mathbf{Z}$ on the subspace.

Computationally cheaper to work with lower-dimensional signal.

Bioucas-Dias et al., IEEE Trans. Geosci. Remote Sens, 2008.

Optimization problem

$$\min_{\mathbf{X}} \quad \underbrace{\frac{1}{2}\|\mathbf{Y}_h - \mathbf{SBXE}\|^2 + \frac{\lambda}{2}\|\mathbf{Y}_m - \mathbf{XER}\|^2}_{\text{Data fidelity term } f(\mathbf{X})} + \underbrace{\tau\phi(\mathbf{X})}_{\text{Regularizer term}}$$

Optimization problem

$$\min_{\mathbf{X}} \quad \underbrace{\frac{1}{2}\|\mathbf{Y}_h - \mathbf{SBXE}\|^2 + \frac{\lambda}{2}\|\mathbf{Y}_m - \mathbf{XER}\|^2}_{\text{Data fidelity term } f(\mathbf{X})} + \underbrace{\tau\phi(\mathbf{X})}_{\text{Regularizer term}}$$

$$\Downarrow \text{Variable splitting}$$

$$\min_{\mathbf{X},\mathbf{V}} \quad f(\mathbf{X}) + \tau\phi(\mathbf{V}) \quad \text{subject to} \quad \mathbf{X} = \mathbf{V}.$$

Plug-and-play framework

ADMM solution

$$\mathbf{X}_0^k = \mathbf{V}^k - \mathbf{U}^k$$
$$\mathbf{V}_0^k = \mathbf{X}^{k+1} + \mathbf{U}^k$$
$$\mu > 0$$

Primal updates:

$$\mathbf{X}^{k+1} = \underset{\mathbf{X}}{\operatorname{argmin}} \;\; f(\mathbf{X}) + \frac{\mu}{2}\|\mathbf{X} - \mathbf{X}_0^k\|^2,$$

$$\mathbf{V}^{k+1} = \underset{\mathbf{V}}{\operatorname{argmin}} \;\; \tau\phi(\mathbf{V}) + \frac{\mu}{2}\|\mathbf{V} - \mathbf{V}_0^k\|^2,$$

Dual update:

$$\mathbf{U}^{k+1} = \mathbf{U}^k + (\mathbf{X}^{k+1} - \mathbf{V}^{k+1}).$$

X update

- $\mathbf{X}^{k+1} = \underset{\mathbf{X}}{\operatorname{argmin}} \;\; \frac{1}{2}\|\mathbf{Y}_h - \mathbf{SBXE}\|^2 + \frac{\lambda}{2}\|\mathbf{Y}_m - \mathbf{XER}\|^2 + \frac{\mu}{2}\|\mathbf{X} - \mathbf{X}_0^k\|^2.$

- Setting gradient to zero results in Sylvester equation,

$$\mathbf{C_1 X} + \mathbf{X C_2} = \mathbf{C_3},$$

$$\begin{aligned}
\mathbf{C_1} &= (\mathbf{SB})^\top \mathbf{SB} \\
\mathbf{C_2} &= \lambda^{-1}(\mathbf{EE}^\top)^{-1}(\mathbf{ER}(\mathbf{ER})^\top \; + \; \mu \mathbf{I}_{\ell_s}) \\
\mathbf{C_3} &= \lambda^{-1}(\lambda(\mathbf{SB})^\top \mathbf{Y}_h \mathbf{E}^\top + \mathbf{Y}_m(\mathbf{ER})^\top + \mathbf{V}_k + \mathbf{U}_k)(\mathbf{EE}^\top)^{-1}
\end{aligned}$$

- Fast Sylvester solver is proposed by Wei et al. (IEEE TIP, 2015).

V update

Definition

Given $f : \mathbb{R}^n \to \mathbb{R}$, proximal operator is defined as,

$$\mathrm{prox}_f(\boldsymbol{x}) = \underset{\boldsymbol{y} \in \mathbb{R}^n}{\mathrm{argmin}} \;\; f(\boldsymbol{y}) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2.$$

V update

Definition

Given $f : \mathbb{R}^n \to \mathbb{R}$, proximal operator is defined as,

$$\mathrm{prox}_f(\boldsymbol{x}) = \operatorname*{argmin}_{\boldsymbol{y} \in \mathbb{R}^n} \; f(\boldsymbol{y}) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2.$$

$$\mathbf{V}^{k+1} = \operatorname*{argmin}_{\mathbf{V}} \; \tau\phi(\mathbf{V}) + \frac{\mu}{2}\|\mathbf{V} - \mathbf{V}_0^k\|^2$$

V update

Definition

Given $f : \mathbb{R}^n \to \mathbb{R}$, proximal operator is defined as,

$$\text{prox}_f(\boldsymbol{x}) = \underset{\boldsymbol{y}\in\mathbb{R}^n}{\text{argmin}} \;\; f(\boldsymbol{y}) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2.$$

$$\mathbf{V}^{k+1} = \underset{\mathbf{V}}{\text{argmin}} \;\; \tau\phi(\mathbf{V}) + \frac{\mu}{2}\left\|\mathbf{V} - \mathbf{V}_0^k\right\|^2 \Rightarrow \mathbf{V}^{k+1} = \text{prox}_{\frac{\tau}{\mu}\phi}(\mathbf{V}_0^k).$$

V update

Definition

Given $f : \mathbb{R}^n \to \mathbb{R}$, proximal operator is defined as,

$$\text{prox}_f(\boldsymbol{x}) = \underset{\boldsymbol{y} \in \mathbb{R}^n}{\text{argmin}} \; f(\boldsymbol{y}) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2.$$

$$\mathbf{V}^{k+1} = \underset{\mathbf{V}}{\text{argmin}} \; \tau\phi(\mathbf{V}) + \frac{\mu}{2}\left\|\mathbf{V} - \mathbf{V}_0^k\right\|^2 \Rightarrow \mathbf{V}^{k+1} = \text{prox}_{\frac{\tau}{\mu}\phi}(\mathbf{V}_0^k).$$

$$\mathbf{V}_0 = \mathbf{V} + \eta$$

$$\eta \sim \mathcal{N}(\mathbf{0}, \frac{\mu}{\tau}\mathrm{I})$$

$$p(\mathbf{V}) = \exp(-\phi(\mathbf{V}))$$

V update

Definition

Given $f : \mathbb{R}^n \to \mathbb{R}$, proximal operator is defined as,

$$\text{prox}_f(\boldsymbol{x}) = \underset{\boldsymbol{y}\in\mathbb{R}^n}{\operatorname{argmin}} \;\; f(\boldsymbol{y}) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2.$$

$$\mathbf{V}^{k+1} = \underset{\mathbf{V}}{\operatorname{argmin}} \;\; \tau\phi(\mathbf{V}) + \frac{\mu}{2}\|\mathbf{V} - \mathbf{V}_0^k\|^2 \Rightarrow \mathbf{V}^{k+1} = \text{prox}_{\frac{\tau}{\mu}\phi}(\mathbf{V}_0^k).$$

$$\Downarrow$$

$$\mathbf{V}^{k+1} = \Psi_{\mu/\tau}(\mathbf{V}_0^k), \text{ where } \Psi \text{ is a denoiser.}$$

$$\mathbf{V}_0 = \mathbf{V} + \eta$$

$$\eta \sim \mathcal{N}(\mathbf{0}, \frac{\mu}{\tau}\mathrm{I})$$

$$p(\mathbf{V}) = \exp(-\phi(\mathbf{V}))$$

V update

Definition

Given $f : \mathbb{R}^n \to \mathbb{R}$, proximal operator is defined as,

$$\text{prox}_f(\boldsymbol{x}) = \underset{\boldsymbol{y} \in \mathbb{R}^n}{\operatorname{argmin}} \; f(\boldsymbol{y}) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2.$$

$$\mathbf{V}^{k+1} = \underset{\mathbf{V}}{\operatorname{argmin}} \; \tau\phi(\mathbf{V}) + \frac{\mu}{2}\|\mathbf{V} - \mathbf{V}_0^k\|^2 \Rightarrow \mathbf{V}^{k+1} = \text{prox}_{\frac{\tau}{\mu}\phi}(\mathbf{V}_0^k).$$

$$\Downarrow$$

$$\mathbf{V}^{k+1} = \Psi_{\mu/\tau}(\mathbf{V}_0^k), \text{ where } \Psi \text{ is a denoiser.}$$

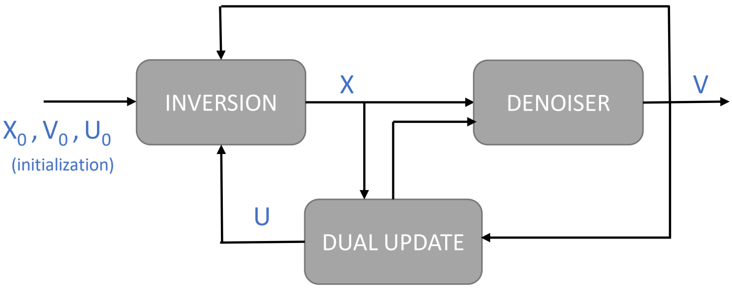
$$\mathbf{V}_0 = \mathbf{V} + \eta$$

$$\eta \sim \mathcal{N}(\mathbf{0}, \frac{\mu}{\tau}\mathrm{I})$$

$$p(\mathbf{V}) = \exp(-\phi(\mathbf{V}))$$

Can we implicitly force a regularizer?

Plug-and-play iterations



Denoising in PnP

Powerful denoisers capturing inter-band correlation

1. DnCNN[Zhang et al, IEEE TIP, 2017]- deep learning based denoiser.

2. BM3D[Dabov et al, IEEE TIP, 2007]- restricted to color images.

3. Non local means[Buades et al, CVPR, 2005]- Good performance but computationally expensive.

4. Sparsity based.[Zhao et al, IEEE TGRS, 2015]

High-dimensional denoising

$$\mathbf{V}^{k+1} = \Psi_{\mu/\tau}(\mathbf{V}_0^k).$$

Challenges

- Every iteration requires a denoising operation.
- Very few denoisers are scalable to hyperspectral images.
- Existing state-of-the-art denoisers are computationally expensive.
- $\{\mathbf{X}_k\}_{k \in \mathbb{Z}^+}$ may not converge to a fixed point $\mathbf{X}^*$.
- $\mathbf{X}^*$ may not be the infimum of any optimization problem.

Optimality conditions?

Primal updates:

$$\mathbf{X}^{k+1} = \operatorname*{argmin}_{\mathbf{X}} \; f(\mathbf{X}) + \frac{\mu}{2}\left\|\mathbf{X} - \mathbf{X}_0^k\right\|^2,$$

$$\mathbf{V}^{k+1} = \Psi_{\mu/\tau}(\mathbf{V}_0^k).$$

Dual update:

$$\mathbf{U}^{k+1} = \mathbf{U}^k + (\mathbf{X}^{k+1} - \mathbf{V}^{k+1}),$$

where Ψ is a denoising operator.

Linear denoiser

Primal updates:

$$\mathbf{X}^{k+1} = \operatorname*{argmin}_{\mathbf{X}} \; f(\mathbf{X}) + \frac{\mu}{2}\|\mathbf{X} - \mathbf{X}_0^k\|^2,$$

$$\mathbf{V}^{k+1} = \mathbf{W}\mathbf{V}_0^k$$

Dual update:

$$\mathbf{U}^{k+1} = \mathbf{U}^k + (\mathbf{X}^{k+1} - \mathbf{V}^{k+1}),$$

where Ψ is a denoising operator.

Convergence guarantees [1]

Plug and Play iterates:

$$\mathbf{X}^{k+1} = \underset{\mathbf{X}}{\operatorname{argmin}} \;\; f(\mathbf{X}) + \frac{\mu}{2}\|\mathbf{X} - \mathbf{X}_0^k\|^2$$

$$\mathbf{V}^{k+1} = \mathbf{W}\mathbf{V}_0^k$$

$$\mathbf{U}^{k+1} = \mathbf{U}^k + (\mathbf{X}^{k+1} - \mathbf{V}^{k+1})$$

[1] Sreehari et al., IEEE TCI, 2016.

Convergence guarantees [1]

Plug and Play iterates:

$$\mathbf{X}^{k+1} = \underset{\mathbf{X}}{\mathrm{argmin}} \quad f(\mathbf{X}) + \frac{\mu}{2}\|\mathbf{X} - \mathbf{X}_0^k\|^2$$

$$\mathbf{V}^{k+1} = \mathbf{W}\mathbf{V}_0^k$$

$$\mathbf{U}^{k+1} = \mathbf{U}^k + (\mathbf{X}^{k+1} - \mathbf{V}^{k+1})$$

$\equiv$

Applying ADMM to:

$$\min_{\mathbf{X},\mathbf{V}} \quad f(\mathbf{X}) + \tau\phi(\mathbf{V})$$

subject to $\quad \mathbf{X} = \mathbf{V}$

[1] Sreehari et al., IEEE TCI, 2016.

Convergence guarantees [1]

Plug and Play iterates:

$$\mathbf{X}^{k+1} = \operatorname*{argmin}_{\mathbf{X}} \; f(\mathbf{X}) + \frac{\mu}{2}\|\mathbf{X} - \mathbf{X}_0^k\|^2$$
$$\mathbf{V}^{k+1} = \mathbf{W}\mathbf{V}_0^k$$
$$\mathbf{U}^{k+1} = \mathbf{U}^k + (\mathbf{X}^{k+1} - \mathbf{V}^{k+1})$$

$\equiv$

Applying ADMM to:

$$\min_{\mathbf{X},\mathbf{V}} \; f(\mathbf{X}) + \tau\phi(\mathbf{V})$$
subject to $\mathbf{X} = \mathbf{V}$

$\Downarrow$

Convergence is well established.

[1] Sreehari et al., IEEE TCI, 2016.

Convergence guarantees [1]

Convegence

Consider the PnP algorithm where the weight matrix for denoising $\mathbf{W}$ is symmetric, PSD and doubly stochastic, then $f(\mathbf{X}^k) + \tau\phi(\mathbf{X}^k)$ converges to the minimum of $f(\mathbf{X}) + \tau\phi(\mathbf{X})$ as $k \uparrow \infty$, where $\phi(\mathbf{X})$ is a convex and quadratic regularizer.

Plug and Play iterates:

$$\mathbf{X}^{k+1} = \operatorname*{argmin}_{\mathbf{X}} \; f(\mathbf{X}) + \frac{\mu}{2}\|\mathbf{X} - \mathbf{X}_0^k\|^2$$

$$\mathbf{V}^{k+1} = \mathbf{W}\mathbf{V}_0^k$$

$$\mathbf{U}^{k+1} = \mathbf{U}^k + (\mathbf{X}^{k+1} - \mathbf{V}^{k+1})$$

$\equiv$

Applying ADMM to:

$$\min_{\mathbf{X},\mathbf{V}} \quad f(\mathbf{X}) + \tau\phi(\mathbf{V})$$

$$\text{subject to} \quad \mathbf{X} = \mathbf{V}$$

$\Downarrow$

Convergence is well established.

[1] Sreehari et al., IEEE TCI, 2016.

High-dimensional denoising methods

High-dimensional denoising

- Given a high-dimensional image $\boldsymbol{f} : \Omega \rightarrow \mathbb{R}^n$ and guide image $\boldsymbol{p} : \Omega \rightarrow \mathbb{R}^\rho$, where $\Omega \subset \mathbb{Z}^d$ is the domain.

- Filtered output $\boldsymbol{g} : \Omega \rightarrow \mathbb{R}^n$:

$$\boldsymbol{g}(\boldsymbol{x}) = \frac{\sum_{\boldsymbol{y} \in W_{\boldsymbol{x}}} \omega(\boldsymbol{x} - \boldsymbol{y}) \; \varphi\big(\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}(\boldsymbol{y})\big) \; \boldsymbol{f}(\boldsymbol{y})}{\sum_{\boldsymbol{y} \in W_{\boldsymbol{x}}} \omega(\boldsymbol{x} - \boldsymbol{y}) \; \varphi\big(\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}(\boldsymbol{y})\big)},$$

- Per-pixel complexity: $\mathcal{O}\big((2S+1)^d(n+\rho)\big)$, S is window radius.

- State-of-the-art fast approximation methods convert non-linear operations to fast convolutions.

---

1) Gastal et al., ACM TOG, 2012.

2) Mozerov et al., IEEE TIP, 2015.

3) Nair et al., IEEE TIP, 2019.

What about convergence?

Sufficient conditions

$\mathbf{W}$ is symmetric, psd and eigenvalues in $[0, 1]$.

What about convergence?

Sufficient conditions

$\mathbf{W}$ is symmetric, psd and eigenvalues in $[0, 1]$.

$$\boldsymbol{g}(\boldsymbol{x}) = \frac{\sum_{\boldsymbol{y} \in W_x} \omega(\boldsymbol{x} - \boldsymbol{y})\ \varphi\big(\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}(\boldsymbol{y})\big)\ \boldsymbol{f}(\boldsymbol{y})}{\sum_{\boldsymbol{y} \in W_x} \omega(\boldsymbol{x} - \boldsymbol{y})\ \varphi\big(\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}(\boldsymbol{y})\big)}.$$

What about convergence?

Sufficient conditions

$\mathbf{W}$ is symmetric, psd and eigenvalues in $[0, 1]$.

$$\boldsymbol{g}(\boldsymbol{x}) = \frac{\sum_{\boldsymbol{y} \in W_x} \omega(\boldsymbol{x} - \boldsymbol{y})\ \varphi\big(\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}(\boldsymbol{y})\big)\ \boldsymbol{f}(\boldsymbol{y})}{\sum_{\boldsymbol{y} \in W_x} \omega(\boldsymbol{x} - \boldsymbol{y})\ \varphi\big(\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}(\boldsymbol{y})\big)}.$$

$$\Downarrow$$

LINEAR FORM :

Let $\mathbf{K}_{\boldsymbol{x},\boldsymbol{y}} = \omega(\boldsymbol{x} - \boldsymbol{y})\ \varphi\big(\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}(\boldsymbol{y})\big)$ and $\mathbf{D}_{\boldsymbol{x},\boldsymbol{x}} = \sum_y \mathbf{K}_{\boldsymbol{x},\boldsymbol{y}}$,

then $\mathbf{W} = \mathbf{D}^{-1}\mathbf{K}$ and $\boldsymbol{g} = \mathbf{W}\boldsymbol{f}$.

What about convergence?

Sufficient conditions

$\mathbf{W}$ is symmetric, psd and eigenvalues in $[0,1]$.

$$\boldsymbol{g}(\boldsymbol{x}) = \frac{\sum_{\boldsymbol{y}\in W_x} \omega(\boldsymbol{x}-\boldsymbol{y})\ \varphi\big(\boldsymbol{p}(\boldsymbol{x})-\boldsymbol{p}(\boldsymbol{y})\big)\ \boldsymbol{f}(\boldsymbol{y})}{\sum_{\boldsymbol{y}\in W_x} \omega(\boldsymbol{x}-\boldsymbol{y})\ \varphi\big(\boldsymbol{p}(\boldsymbol{x})-\boldsymbol{p}(\boldsymbol{y})\big)}.$$

$$\Downarrow$$

LINEAR FORM :

Let $\mathbf{K}_{\boldsymbol{x},\boldsymbol{y}} = \omega(\boldsymbol{x}-\boldsymbol{y})\ \varphi\big(\boldsymbol{p}(\boldsymbol{x})-\boldsymbol{p}(\boldsymbol{y})\big)$ and $\mathbf{D}_{\boldsymbol{x},\boldsymbol{x}} = \sum_{y} \mathbf{K}_{\boldsymbol{x},\boldsymbol{y}}$,

then $\mathbf{W} = \mathbf{D}^{-1}\mathbf{K}$ and $\boldsymbol{g} = \mathbf{W}\boldsymbol{f}$.

$\mathbf{W}$ is not even symmetric.

DSG-NLM

$$\mathbf{K}_{x,y} = \varphi(\boldsymbol{p}_x - \boldsymbol{p}_y),$$

Derivation of weight matrix from $\mathbf{K}$:

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is separable hat function,}$$

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x\in\Omega_y}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y\in\Omega_x}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

and

$$\mathbf{W}_{x,y} = \alpha\mathbf{H}_{x,y}, \quad \alpha = \Big(\max_x \sum_{y\in\Omega_x}\mathbf{H}_{x,y}\Big)^{-1},$$

$$\mathbf{W}_{x,x} \leftarrow \mathbf{W}_{x,x} + 1 - \sum_{y\in\Omega_x}\mathbf{W}_{x,y}.$$

Sreehari et al., IEEE TCI, 2016.

DSG-NLM

$$\mathbf{K}_{x,y} = \varphi(\boldsymbol{p}_x - \boldsymbol{p}_y),$$

positive definite range kernel

Derivation of weight matrix from $\mathbf{K}$:

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is separable hat function,}$$

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x\in\Omega_y}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y\in\Omega_x}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

and

$$\mathbf{W}_{x,y} = \alpha\mathbf{H}_{x,y}, \quad \alpha = \Big(\max_x \sum_{y\in\Omega_x}\mathbf{H}_{x,y}\Big)^{-1},$$

$$\mathbf{W}_{x,x} \leftarrow \mathbf{W}_{x,x} + 1 - \sum_{y\in\Omega_x}\mathbf{W}_{x,y}.$$

Sreehari et al., IEEE TCI, 2016.

DSG-NLM

$$\mathbf{K}_{x,y} = \varphi(\boldsymbol{p}_x - \boldsymbol{p}_y),$$

Derivation of weight matrix from $\mathbf{K}$:

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is separ}$$

Multiply by a spatial kernel

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x\in\Omega_y}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y\in\Omega_x}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

and

$$\mathbf{W}_{x,y} = \alpha\mathbf{H}_{x,y}, \quad \alpha = \Big(\max_x \sum_{y\in\Omega_x}\mathbf{H}_{x,y}\Big)^{-1},$$

$$\mathbf{W}_{x,x} \leftarrow \mathbf{W}_{x,x} + 1 - \sum_{y\in\Omega_x}\mathbf{W}_{x,y}.$$

Sreehari et al., IEEE TCI, 2016.

DSG-NLM

$$\mathbf{K}_{x,y} = \varphi(\boldsymbol{p}_x - \boldsymbol{p}_y),$$

Derivation of weight matrix from $\mathbf{K}$:

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is separable hat function,}$$

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x\in\Omega_y}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y\in\Omega_x}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

Normalize the weight matrix

and

$$\mathbf{W}_{x,y} = \alpha\mathbf{H}_{x,y}, \quad \alpha = \Big(\max_x \sum_{y\in\Omega_x}\mathbf{H}_{x,y}\Big),$$

$$\mathbf{W}_{x,x} \leftarrow \mathbf{W}_{x,x} + 1 - \sum_{y\in\Omega_x}\mathbf{W}_{x,y}.$$

Sreehari et al., IEEE TCI, 2016.

DSG-NLM

$$\mathbf{K}_{x,y} = \varphi(\boldsymbol{p}_x - \boldsymbol{p}_y),$$

Derivation of weight matrix from $\mathbf{K}$:

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is separable hat function},$$

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x\in\Omega_y}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y\in\Omega_x}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

and

$$\mathbf{W}_{x,y} = \alpha\mathbf{H}_{x,y}, \quad \alpha = \Big(\max_x \sum_{y\in\Omega_x}\mathbf{H}_{x,y}\Big)^{-1},$$

$$\mathbf{W}_{x,x} \leftarrow \mathbf{W}_{x,x} + 1 - \sum_{y\in\Omega_x}\mathbf{W}_{x,y}.$$

Make the weight matrix doubly stochastic

Sreehari et al., IEEE TCI, 2016.

DSG-NLM

$$\mathbf{K}_{x,y} = \varphi(\boldsymbol{p}_x - \boldsymbol{p}_y),$$

Derivation of weight matrix from $\mathbf{K}$:

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is separable hat function,}$$

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x\in\Omega_y}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y\in\Omega_x}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

and

Computationally expensive!

$$\mathbf{W}_{x,y} = \alpha\mathbf{H}_{x,y}, \quad \alpha = \Big(\max_x \sum_{y\in\Omega_x}\mathbf{H}_{x,y}\Big)^{-1},$$

$$\mathbf{W}_{x,x} \leftarrow \mathbf{W}_{x,x} + 1 - \sum_{y\in\Omega_x}\mathbf{W}_{x,y}.$$

3 minutes to denoise $256 \times 256 \times 128$ image!

Sreehari et al., IEEE TCI, 2016.

Unni et al., GlobalSIP, 2018.

Proposed Denoiser

Proposed kernel

Approximation

$$\mathbf{K}_{x,y} = \varphi(\boldsymbol{p}_x - \boldsymbol{p}_y) \approx \sum_{\ell=1}^{m_0} \varphi(\boldsymbol{p}_x - \boldsymbol{\mu}_\ell)\varphi(\boldsymbol{p}_y - \boldsymbol{\mu}_\ell),$$

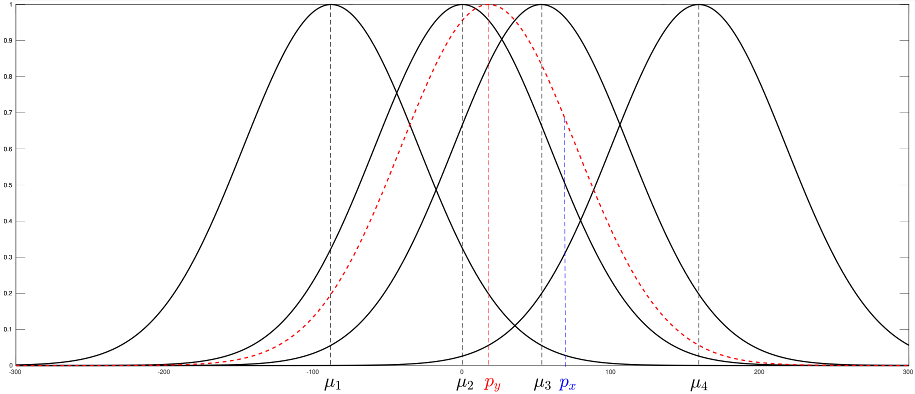
where $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{m_0}$ are the centroids of the m_0 clusters formed by partitioning the range space.

Proposed kernel

Approximation

$$\mathbf{K}_{x,y} = \varphi(\boldsymbol{p}_x - \boldsymbol{p}_y) \approx \sum_{\ell=1}^{m_0} \varphi(\boldsymbol{p}_x - \boldsymbol{\mu}_\ell)\varphi(\boldsymbol{p}_y - \boldsymbol{\mu}_\ell),$$

where $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{m_0}$ are the centroids of the m_0 clusters formed by partitioning the range space.



Proposed kernel

Approximation

$$\mathbf{K}_{x,y} = \varphi(\boldsymbol{p}_x - \boldsymbol{p}_y) \approx \sum_{\ell=1}^{m_0} \varphi(\boldsymbol{p}_x - \boldsymbol{\mu}_\ell)\varphi(\boldsymbol{p}_y - \boldsymbol{\mu}_\ell),$$

where $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{m_0}$ are the centroids of the m_0 clusters formed by partitioning $\mathfrak{R} = \{\boldsymbol{p}_x : x \in \Omega\}$.

Proposition

The matrix $\mathbf{K}$ is nonnegative, symmetric, and positive semidefinite.

Forcing eigenvalues in [0,1]

$$\mathbf{K}_{x,y} = \sum_{\ell=1}^{m_0} \varphi(\boldsymbol{p}_x - \boldsymbol{\mu}_\ell)\varphi(\boldsymbol{p}_y - \boldsymbol{\mu}_\ell),$$

Derivation of weight matrix from $\mathbf{K}$:

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is seperable hat function},$$

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x\in\Omega_y}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y\in\Omega_x}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

and

$$\mathbf{W}_{x,y} = \alpha\mathbf{H}_{x,y}, \quad \alpha = \left(\max_x \sum_{y\in\Omega_x}\mathbf{H}_{x,y}\right)^{-1},$$

$$\mathbf{W}_{x,x} \leftarrow \mathbf{W}_{x,x} + 1 - \sum_{y\in\Omega_x}\mathbf{W}_{x,y}.$$

$$\mathbf{K}_{x,y} = \sum_{\ell=1}^{m_0} \varphi(\boldsymbol{p}_x - \boldsymbol{\mu}_\ell)\varphi(\boldsymbol{p}_y - \boldsymbol{\mu}_\ell),$$

Derivation of weight matrix from $\mathbf{K}$:

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is seperable hat function,}$$

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x\in\Omega_y}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y\in\Omega_x}\mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

and

$$\alpha = \Big(\max_x \sum_{y\in\Omega_x}\mathbf{H}_{x,y}\Big)^{-1},$$

$$(\mathbf{WX})_x = \mathbf{X}_x + \alpha\sum_{y\in\Omega_x}\mathbf{H}_{x,y}\mathbf{X}_y - \alpha\Big(\sum_{y\in\Omega_x}\mathbf{H}_{x,y}\Big)\mathbf{X}_x.$$

$$\mathbf{K}_{x,y} = \sum_{\ell=1}^{m_0} \varphi(\boldsymbol{p}_x - \boldsymbol{\mu}_\ell)\varphi(\boldsymbol{p}_y - \boldsymbol{\mu}_\ell),$$

Derivation of weight matrix from $\mathbf{K}$:

Let $\eta_x = \sum_{y \in \Omega_x} \mathbf{G}_{x,y}$

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is seperable hat function,}$$

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x \in \Omega_y} \mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y \in \Omega_x} \mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

and

$$\alpha = \Big(\max_x \sum_{y \in \Omega_x} \mathbf{H}_{x,y}\Big)^{-1},$$

$$(\mathbf{WX})_x = \mathbf{X}_x + \alpha \sum_{y \in \Omega_x} \mathbf{H}_{x,y}\mathbf{X}_y - \alpha\Big(\sum_{y \in \Omega_x} \mathbf{H}_{x,y}\Big)\mathbf{X}_x.$$

$$\mathbf{K}_{x,y} = \sum_{\ell=1}^{m_0} \varphi(\boldsymbol{p}_x - \boldsymbol{\mu}_\ell)\varphi(\boldsymbol{p}_y - \boldsymbol{\mu}_\ell),$$

Derivation of weight matrix from $\mathbf{K}$:

Let $\eta_x = \sum_{y\in\Omega_x} \mathbf{G}_{x,y}$

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is seperable hat function,}$$

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x\in\Omega_y} \mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y\in\Omega_x} \mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

and

$$\alpha = \Big(\max_x \sum_{y\in\Omega_x} \mathbf{H}_{x,y}\Big)^{-1},$$

$\mathbf{H}_{x,y} = \frac{\mathbf{G}_{x,y}}{\sqrt{\eta_x}\sqrt{\eta_y}}$

$$(\mathbf{WX})_x = \mathbf{X}_x + \alpha \sum_{y\in\Omega_x} \mathbf{H}_{x,y}\mathbf{X}_y - \alpha\Big(\sum_{y\in\Omega_x} \mathbf{H}_{x,y}\Big)\mathbf{X}_x.$$

Operator based implementation

$$\mathbf{K}_{x,y} = \sum_{\ell=1}^{m_0} \varphi(\boldsymbol{p}_x - \boldsymbol{\mu}_\ell)\varphi(\boldsymbol{p}_y - \boldsymbol{\mu}_\ell),$$

Derivation of weight matrix from $\mathbf{K}$:

Let $\eta_x = \sum_{y \in \Omega_x} \mathbf{G}_{x,y}$

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is seperable hat function},$$

$$\mathbf{H}_{x,y} = \mathbf{G}_{x,y}\Big(\sum_{x \in \Omega_y} \mathbf{G}_{x,y}\Big)^{-\frac{1}{2}}\Big(\sum_{y \in \Omega_x} \mathbf{G}_{x,y}\Big)^{-\frac{1}{2}},$$

$$\mathbf{H}_{x,y} = \frac{\mathbf{G}_{x,y}}{\sqrt{\eta_x}\sqrt{\eta_y}}$$

and

$$\alpha = \Big(\max_x \sum_{y \in \Omega_x} \mathbf{H}_{x,y}\Big)^{-1},$$

$$(\mathbf{WX})_x = \mathbf{X}_x + \alpha \sum_{y \in \Omega_x} \mathbf{H}_{x,y}\mathbf{X}_y - \alpha\Big(\sum_{y \in \Omega_x} \mathbf{H}_{x,y}\Big)\mathbf{X}_x.$$

Operator based implementation

$$\mathbf{K}_{x,y} = \sum_{\ell=1}^{m_0} \varphi(\boldsymbol{p}_x - \boldsymbol{\mu}_\ell)\varphi(\boldsymbol{p}_y - \boldsymbol{\mu}_\ell),$$

Derivation of weight matrix from $\mathbf{K}$:

Let $\eta_x = \sum_{y \in \Omega_x} \mathbf{G}_{x,y}$

$$\mathbf{G}_{x,y} = \Lambda\left(\frac{x-y}{N+1}\right)\mathbf{K}_{x,y}, \quad \Lambda(x) \text{ is seperable hat function,}$$

Just fast convolutions and matrix-vector multiplications required.

and

$$\mathbf{H}_{x,y} = \frac{\mathbf{G}_{x,y}}{\sqrt{\eta_x}\sqrt{\eta_y}}$$

$$\alpha = \left(\max_x \sum_{y \in \Omega_x} \mathbf{H}_{x,y}\right)^{-1},$$

$$(\mathbf{W}\mathbf{X})_x = \mathbf{X}_x + \alpha \sum_{y \in \Omega_x} \mathbf{H}_{x,y}\mathbf{X}_y - \alpha\left(\sum_{y \in \Omega_x} \mathbf{H}_{x,y}\right)\mathbf{X}_x.$$

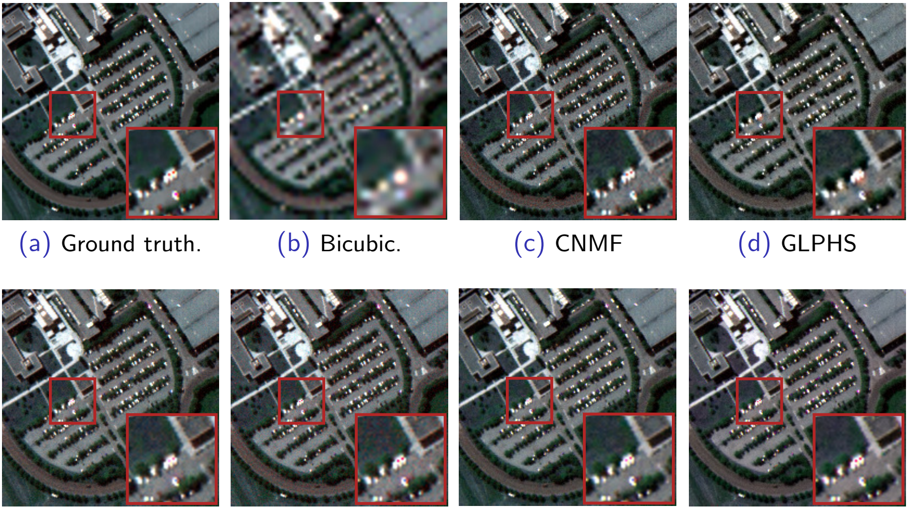
Advantages of the proposed denoiser

- Per-pixel complexity: $\mathcal{O}(m_0(n+\rho))$.
- Effective speedup w.r.t brute-force implementations: $(2S+1)^d/m_0$.
- Applicable in iterative PnP frameworks for real-time applications.
- Convergence is guaranteed for PnP-ADMM.

Computational speed-up

Search Size	Proposed	Brute-force
9×9	3.80	99.60
11×11	3.75	159.45
15×15	3.76	349.90
17×17	3.78	510.70
19×19	3.83	805.10

Table: Timings (sec) of the direct and fast implementations of NLM for $540 \times 420 \times 128$ image for different search size with fixed patch size of 3×3.



(a) Ground truth. (b) Bicubic. (c) CNMF (d) GLPHS

(e) R-FUSE (f) Sparse (g) HySURE (h) PnP-FUSION.

Figure: Comparison of fusion results for the Pavia dataset of size $200 \times 200 \times 93$. Time taken: 17 seconds for 15 iterations.

	Pavia				Paris			
Methods	RMSE	ERGAS	SAM	UIQI	RMSE	ERGAS	SAM	UIQI
CNMF	0.026	4.089	4.420	0.957	0.061	5.402	5.899	0.756
GLPHS	0.027	4.005	5.319	0.957	0.052	4.754	4.736	0.812
SPARSE	0.013	2.166	3.817	0.983	**0.045**	**4.047**	2.983	**0.856**
R-FUSE	0.014	2.083	3.600	0.983	0.048	4.363	3.390	0.823
HySURE	**0.012**	**1.864**	**3.160**	**0.987**	0.048	4.290	3.772	0.818
PnP-FUSION	**0.012**	1.966	3.512	**0.987**	**0.045**	4.119	**2.856**	0.839

	Chikusei			
Methods	RMSE	ERGAS	SAM	UIQI
CNMF	0.015	3.510	3.788	0.894
GLPHS	0.022	4.204	5.079	0.845
SPARSE	0.012	4.893	3.532	0.881
R-FUSE	0.010	3.162	2.873	0.906
HySURE	0.010	3.878	2.974	0.902
PnP-FUSION	**0.009**	**3.097**	**2.497**	**0.922**

Table: Performance comparison for three datasets using standard quality metrics.

Conclusions

- Proposed a fast efficient high-dimensional kernel denoiser.
- Plug-and-play (PnP) iterations with the proposed denoiser converge.
- Results on HS-MS fusion demonstrated.

Open problems

- Convergence of PnP iterations when denoiser is not a proximal map.
- Powerful denoisers with convergence guarantess.
- Other image restoration techniques.

This work was supported by EMR grant SERB/F/6047/2016-2017 from DST-SERB, Government of India.

References 1

- E. S. Gastal and M. M. Oliveira, "Adaptive manifolds for real-time high- dimensional filtering," ACM Transactions on Graphics, vol. 31, no. 4, pp. 33:1-33:13, 2012
- K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising with block-matching and 3D filtering," Proc. SPIE Electronic Imaging, vol. 6064, no. 30, pp. 1-12, 2006.
- M. Simes, J. M. Bioucas-Dias, L. B. Almeida, and J. Chanus- sot, "A convex formulation for hyperspectral image superresolution via subspace-based regularization," IEEE Transactions on Image Processing, vol. 53, no. 6, pp. 3373-3388, 2015.
- S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," Foundations and Trends in Machine learning, vol. 3, no. 1, pp. 1-122, 2011.
- M. Selva, B. Aiazzi, F. Butera, L. Chiarantini, and S. Baronti, "Hyper-sharpening: A first approach on SIM-GA data," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 8, no. 6, pp. 3008-3024, 2015.
- Y. Sun, B. Wohlberg, and U. S. Kamilov, "An online plug-and-play algorithm for regularized image reconstruction,", IEEE Transactions on Computational Imaging, 2019.
- Y. Romano, M. Elad, and P. Milanfar, "The little engine that could regularization by denoising (RED)", SIAM Journal of Imaging Sciences, vol. 10, no. 4, pp. 1804-1844, 2017.

References 2

- S. Sreehari, S. V. Venkatakrishnan, B. Wohlberg, G. T. Buzzard, L. F. Drummy, J. P. Simmons, and C. A. Bouman, "Plug- and-play priors for bright field electron tomography and sparse interpolation," IEEE Transactions on Computational Imaging, vol. 2, no. 4, pp. 408-423, 2016.
- N. Yokoya, C. Grohnfeldt, and J. Chanussot, "Hyperspectral and multispectral data fusion: A comparative review of the recent literature," IEEE Geoscience Remote Sensing Magazine, vol. 5, no. 2, pp. 29-56, 2017.
- Q. Wei, N. Dobigeon, and J. Y. Tourneret, "Fast fusion of multi- band images based on solving a sylvester equation," IEEE Transactions on Computational Imaging, vol. 24, no. 11, pp. 4109-4121, 2015.
- Q. Wei, J. M. Bioucas-Dias, N. Dobigeon, and J. Y. Tourneret, "Hyperspectral and multispectral image fusion based on a sparse representation," IEEE Transactions on Image Processing, vol. 53, no. 7, pp. 3658-3668, 2015.
- N. Yokoya, T. Yairi, and A. Iwasaki, "Coupled nonnegative ma- trix factorization unmixing for hyperspectral and multispectral data fusion," IEEE Transactions on Image Processing, vol. 50, no. 2, pp. 528-537, 2012.
- E. T. Reehorst and P. Schniter, "Regularization by Denoising: Clarifications and New Interpretations", IEEE Transactions on Computational Imaging, vol. 5, no. 1, pp. 52-67, 2018.

Thanks for listening!