Solving RED with Weighted Proximal Methods

Tao Hong
Joint work with Irad Yavneh and Michael Zibulevsky

Computer Science Department
Technion - Israel Institute of Technology

Outline

Inverse Problems – Optimization Problem

Image Denoising - “Simplest” Inverse:

$$\underbrace{\boldsymbol{y}}_{\text{Measured}} = \underbrace{\boldsymbol{x}}_{\text{Clean}} + \underbrace{\boldsymbol{n}}_{\text{Noise (AWGN)}}$$

Inverse Problems – Optimization Problem

Image Denoising - "Simplest" Inverse:

$$\underbrace{\boldsymbol{y}}_{\text{Measured}} = \underbrace{\boldsymbol{x}}_{\text{Clean}} + \underbrace{\boldsymbol{n}}_{\text{Noise (AWGN)}}$$

How?

Inverse Problems – Optimization Problem

Image Denoising - "Simplest" Inverse:

$$\underbrace{\boldsymbol{y}}_{\text{Measured}} = \underbrace{\boldsymbol{x}}_{\text{Clean}} + \underbrace{\boldsymbol{n}}_{\text{Noise (AWGN)}}$$

How?

Maximum a Posteriori Probability (MAP) – Prior

Optimization Problem

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \underbrace{\ell(\boldsymbol{x}, \boldsymbol{y})}_{\text{Data Fidelity}} + \lambda \underbrace{\rho(\boldsymbol{x})}_{\text{Prior}}$$

$\ell(\boldsymbol{x}, \boldsymbol{y})$: linear or nonlinear $\rightarrow \frac{1}{2\sigma^2}\|\boldsymbol{x} - \boldsymbol{y}\|_2^2$ or $\frac{1}{2\sigma^2}\|\boldsymbol{Hx} - \boldsymbol{y}\|_2^2$

$\rho(\cdot)$: TV, sparsity, low-rank, CNN etc.

Inverse Problems – Optimization Problem

Image Denoising - "Simplest" Inverse:

$$\underbrace{\boldsymbol{y}}_{\text{Measured}} = \underbrace{\boldsymbol{x}}_{\text{Clean}} + \underbrace{\boldsymbol{n}}_{\text{Noise (AWGN)}}$$

How?

Maximum a Posteriori Probability (MAP) – Prior

Optimization Problem

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \underbrace{\ell(\boldsymbol{x}, \boldsymbol{y})}_{\text{Data Fidelity}} + \lambda \underbrace{\rho(\boldsymbol{x})}_{\text{Prior}}$$

$\ell(\boldsymbol{x}, \boldsymbol{y})$: linear or nonlinear $\rightarrow \frac{1}{2\sigma^2}\|\boldsymbol{x} - \boldsymbol{y}\|_2^2$ or $\frac{1}{2\sigma^2}\|\boldsymbol{Hx} - \boldsymbol{y}\|_2^2$

$\rho(\cdot)$: TV, sparsity, low-rank, CNN etc.

Finding effective $\rho(\cdot)$?

What is RED? – REgularization by Denoising

image denoising

About 123'000 results (**0.05** sec)

A non-local algorithm for **image denoising**

A Buades, B Coll, JM Morel - 2005 IEEE Computer Society …, 2005 - ieeexplore.ieee.org

We propose a new measure, the method noise, to evaluate and compare the performance of digital **image denoising** methods. We first compute and analyze this method noise for a wide class of **denoising** algorithms, namely the local smoothing filters. Second, we propose a new …

Cited by 4996 Related articles All 24 versions

Can we utilize these denoising algorithms as the prior? and How?

What is RED? – REgularization by Denoising

image denoising

About 123'000 results (**0.05** sec)

A non-local algorithm for **image denoising**
A Buades, B Coll, JM Morel - 2005 IEEE Computer Society …, 2005 - ieeexplore.ieee.org
We propose a new measure, the method noise, to evaluate and compare the performance of digital **image denoising** methods. We first compute and analyze this method noise for a wide class of **denoising** algorithms, namely the local smoothing filters. Second, we propose a new …
Cited by 4996 Related articles All 24 versions

Can we utilize these denoising algorithms as the prior? and How?

semi-positive

What is RED? – REgularization by Denoising

Can we utilize these denoising algorithms as the prior? and How?

semi-positive

Romano et al. [REM17] $\rightarrow$ RED:

$$\rho(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$$

$\boldsymbol{f}(\boldsymbol{x})$: abstract image denoising algorithms

What is RED? – REgularization by Denoising

Can we utilize these denoising algorithms as the prior? and How?

semi-positive

Romano et al. [REM17] $\rightarrow$ RED:

$$\rho(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$$

$\boldsymbol{f}(\boldsymbol{x})$: abstract image denoising algorithms

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \phi(\boldsymbol{x}) \triangleq \ell(\boldsymbol{x}, \boldsymbol{y}) + \frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$$

What is RED? – REgularization by Denoising

Can we utilize these denoising algorithms as the prior? and How?

semi-positive

Romano et al. [REM17] $\rightarrow$ RED:

$$\rho(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$$

$\boldsymbol{f}(\boldsymbol{x})$: abstract image denoising algorithms

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \phi(\boldsymbol{x}) \triangleq \ell(\boldsymbol{x}, \boldsymbol{y}) + \frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$$

How to minimize $\phi(\boldsymbol{x})$? It is weird $\rightarrow \boldsymbol{f}(\boldsymbol{x})$.

The Properties of RED

Assumptions:

- Differentiability: $\boldsymbol{f}(\boldsymbol{x}) : [0,1]^n \rightarrow [0,1]^n$
- Local Homogeneity: $\boldsymbol{f}(c\boldsymbol{x}) = c\boldsymbol{f}(\boldsymbol{x}),$ if $|c-1| \leq \varepsilon \ll 1$
- Passivity: $\|\boldsymbol{f}(\boldsymbol{x})\| \leq \|\boldsymbol{x}\|$

The Properties of RED

Assumptions:

- Differentiability: $\boldsymbol{f}(\boldsymbol{x}) : [0,1]^n \to [0,1]^n$
- Local Homogeneity: $\boldsymbol{f}(c\boldsymbol{x}) = c\boldsymbol{f}(\boldsymbol{x}),\ \text{if}\ |c-1| \le \varepsilon \ll 1$
- Passivity: $\|\boldsymbol{f}(\boldsymbol{x})\| \le \|\boldsymbol{x}\|$

$$\begin{aligned}\nabla_{\boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x}) \cdot \boldsymbol{x} &= \lim_{\varepsilon -> 0} \frac{\boldsymbol{f}(\boldsymbol{x}+\varepsilon\boldsymbol{x})-\boldsymbol{f}(\boldsymbol{x})}{\varepsilon} \\ &= \lim_{\varepsilon -> 0} \frac{(1+\varepsilon)\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{f}(\boldsymbol{x})}{\varepsilon} \\ &= \boldsymbol{f}(\boldsymbol{x})\end{aligned}$$

$$\frac{\partial\left(\frac{1}{2}\boldsymbol{x}^{\mathcal{T}}\left(\boldsymbol{x}-\boldsymbol{f}(\boldsymbol{x})\right)\right)}{\partial \boldsymbol{x}} = \boldsymbol{x} - \frac{1}{2}\boldsymbol{f}(\boldsymbol{x}) - \frac{1}{2}\nabla_{\boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x}) \cdot \boldsymbol{x} = \boxed{\boldsymbol{x}-\boldsymbol{f}(\boldsymbol{x})}$$

$$\frac{\partial^2\left(\frac{1}{2}\boldsymbol{x}^{\mathcal{T}}\left(\boldsymbol{x}-\boldsymbol{f}(\boldsymbol{x})\right)\right)}{\partial \boldsymbol{x}\partial \boldsymbol{x}^{\mathcal{T}}} \succeq 0$$

The Properties of RED

Assumptions:

- Differentiability: $\boldsymbol{f}(\boldsymbol{x}) : [0,1]^n \rightarrow [0,1]^n$
- Local Homogeneity: $\boldsymbol{f}(c\boldsymbol{x}) = c\boldsymbol{f}(\boldsymbol{x}), \text{ if } |c-1| \leq \varepsilon \ll 1$
- Passivity: $\|\boldsymbol{f}(\boldsymbol{x})\| \leq \|\boldsymbol{x}\|$

$$\begin{aligned}
\nabla_{\boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}) \cdot \boldsymbol{x} &= \lim_{\varepsilon -> 0} \frac{\boldsymbol{f}(\boldsymbol{x}+\varepsilon\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x})}{\varepsilon} \\
&= \lim_{\varepsilon -> 0} \frac{(1+\varepsilon)\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x})}{\varepsilon} \\
&= \boldsymbol{f}(\boldsymbol{x})
\end{aligned}$$

$$\frac{\partial\left(\frac{1}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))\right)}{\partial \boldsymbol{x}} = \boldsymbol{x} - \frac{1}{2}\boldsymbol{f}(\boldsymbol{x}) - \frac{1}{2}\nabla_{\boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x}) \cdot \boldsymbol{x} = \boxed{\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})}$$

$$\frac{\partial^2\left(\frac{1}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))\right)}{\partial \boldsymbol{x} \partial \boldsymbol{x}^{\mathcal{T}}} \succeq 0$$

Conclusions:

- $\rho(\boldsymbol{x})$ is convex. If $\ell(\boldsymbol{x}, \boldsymbol{y})$ is convex, $\phi(\boldsymbol{x})$ is convex.
- Evaluate one time gradient or $\phi(\boldsymbol{x})$, call one time $\boldsymbol{f}(\boldsymbol{x})$.

How Many Algorithms Satisfy these Assumptions?

[REM17]: We have many, some of them are state-of-the-art.

NLM, Bilateral, kernal regression, TNRD etc.

Others ε-modified: Median, K-svd, BM3D, EPLL, CNN etc.

Solvers in RED – $\ell(\boldsymbol{x}, \boldsymbol{y}) + \frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$

- gradient based methods: gradient descent/Nesterov Acceleration, conjugate gradient, BFGS, LBFGS etc. – line search? [NW06]

Solvers in RED – $\ell(\boldsymbol{x}, \boldsymbol{y}) + \frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$

- gradient based methods: gradient descent/Nesterov Acceleration, conjugate gradient, BFGS, LBFGS etc. – line search? [NW06]
- ADMM [BPC$^+$11]

Solvers in RED – $\ell(\boldsymbol{x}, \boldsymbol{y}) + \frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$

- gradient based methods: gradient descent/Nesterov Acceleration, conjugate gradient, BFGS, LBFGS etc. – line search? [NW06]
- ADMM [BPC$^+$11]
- fixed-point (FP) [REM17]

$$\frac{1}{\sigma^2}\boldsymbol{H}^{\mathcal{T}}(\boldsymbol{H}\boldsymbol{x}_{k+1} - \boldsymbol{y}) + \lambda(\boldsymbol{x}_{k+1} - \boldsymbol{f}(\boldsymbol{x}_k)) = 0 \qquad \text{Fourier or CG}$$

Solvers in RED – $\ell(\boldsymbol{x}, \boldsymbol{y}) + \frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$

- gradient based methods: gradient descent/Nesterov Acceleration, conjugate gradient, BFGS, LBFGS etc. – line search? [NW06]
- ADMM [BPC+11]
- fixed-point (FP) [REM17]

$$\frac{1}{\sigma^2}\boldsymbol{H}^{\mathcal{T}}(\boldsymbol{H}\boldsymbol{x}_{k+1} - \boldsymbol{y}) + \lambda(\boldsymbol{x}_{k+1} - \boldsymbol{f}(\boldsymbol{x}_k)) = 0 \qquad \text{Fourier or CG}$$

- Vector Extrapolation (VE) [HRE19]

Solvers in RED – $\ell(\boldsymbol{x},\boldsymbol{y}) + \frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$

- gradient based methods: gradient descent/Nesterov Acceleration, conjugate gradient, BFGS, LBFGS etc. – line search? [NW06]
- ADMM [BPC$^+$11]
- fixed-point (FP) [REM17]

$$\frac{1}{\sigma^2}\boldsymbol{H}^{\mathcal{T}}(\boldsymbol{H}\boldsymbol{x}_{k+1} - \boldsymbol{y}) + \lambda(\boldsymbol{x}_{k+1} - \boldsymbol{f}(\boldsymbol{x}_k)) = 0 \qquad \text{Fourier or CG}$$

- Vector Extrapolation (VE) [HRE19]
- Accelerated Proximal Gradient (APG) [RS19]
- ...

In practice: APG $\geq$ VE $>$ FP $>$ ADMM $>$ gradient based

Solvers in RED – $\ell(\boldsymbol{x}, \boldsymbol{y}) + \frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$

- gradient based methods: gradient descent/Nesterov Acceleration, conjugate gradient, BFGS, LBFGS etc. – line search? [NW06]
- ADMM [BPC$^+$11]
- fixed-point (FP) [REM17]

$$\frac{1}{\sigma^2}\boldsymbol{H}^{\mathcal{T}}(\boldsymbol{H}\boldsymbol{x}_{k+1} - \boldsymbol{y}) + \lambda(\boldsymbol{x}_{k+1} - \boldsymbol{f}(\boldsymbol{x}_k)) = 0 \qquad \text{Fourier or CG}$$

- Vector Extrapolation (VE) [HRE19]
- Accelerated Proximal Gradient (APG) [RS19]
- ...

In practice: APG $\geq$ VE $>$ FP $>$ ADMM $>$ gradient based

But, but, but

Solvers in RED – $\ell(\boldsymbol{x},\boldsymbol{y}) + \frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$

- gradient based methods: gradient descent/Nesterov Acceleration, conjugate gradient, BFGS, LBFGS etc. – line search? [NW06]
- ADMM [BPC$^+$11]
- fixed-point (FP) [REM17]

$$\frac{1}{\sigma^2}\boldsymbol{H}^{\mathcal{T}}(\boldsymbol{H}\boldsymbol{x}_{k+1} - \boldsymbol{y}) + \lambda(\boldsymbol{x}_{k+1} - \boldsymbol{f}(\boldsymbol{x}_k)) = 0 \qquad \text{Fourier or CG}$$

- Vector Extrapolation (VE) [HRE19]
- Accelerated Proximal Gradient (APG) [RS19]
- ...

In practice: APG $\geq$ VE $>$ FP $>$ ADMM $>$ gradient based

But, but, but

Weighted Proximal Methods can do better. :-)

Development of the Proximal Gradient Methods

Composite problem:

$$\min_{\boldsymbol{x}} \phi(\boldsymbol{x}) \triangleq g(\boldsymbol{x}) + h(\boldsymbol{x})$$

$g(\boldsymbol{x})$: convex, differentiability

$h(\boldsymbol{x})$: convex, can be nonsmooth

The solution is nonempty.

[1] Euclidean distance here. Bregman distance in general, [Bec17].

Development of the Proximal Gradient Methods

Composite problem:

$$\min_{\boldsymbol{x}} \phi(\boldsymbol{x}) \triangleq g(\boldsymbol{x}) + h(\boldsymbol{x})$$

$g(\boldsymbol{x})$: convex, differentiability
$h(\boldsymbol{x})$: convex, can be nonsmooth
The solution is nonempty.

$\boldsymbol{x}_k$: solution at kth iteration
Linearizing $g(\boldsymbol{x})$ at $\boldsymbol{x}_k$ [1]:

$$g(\boldsymbol{x})+h(\boldsymbol{x}) \leq \hat{\phi}(\boldsymbol{x},\boldsymbol{x}_k) \triangleq g(\boldsymbol{x}_k) + < \nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k), \boldsymbol{x}-\boldsymbol{x}_k > + \frac{L}{2}\|\boldsymbol{x}-\boldsymbol{x}_k\|_2^2 + h(\boldsymbol{x})$$

$$\nabla^2_{\boldsymbol{x}} g(\boldsymbol{x}) \preceq L$$

[1] Euclidean distance here. Bregman distance in general, [Bec17].

Development of the Proximal Gradient Methods

Composite problem:

$$\min_{\boldsymbol{x}} \phi(\boldsymbol{x}) \triangleq g(\boldsymbol{x}) + h(\boldsymbol{x})$$

$g(\boldsymbol{x})$: convex, differentiability
$h(\boldsymbol{x})$: convex, can be nonsmooth
The solution is nonempty.

$\boldsymbol{x}_k$: solution at kth iteration
Linearizing $g(\boldsymbol{x})$ at $\boldsymbol{x}_k$ [1]:

$$g(\boldsymbol{x}) + h(\boldsymbol{x}) \leq \hat{\phi}(\boldsymbol{x}, \boldsymbol{x}_k) \triangleq g(\boldsymbol{x}_k) + < \nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k), \boldsymbol{x} - \boldsymbol{x}_k > + \frac{L}{2} \|\boldsymbol{x} - \boldsymbol{x}_k\|_2^2 + h(\boldsymbol{x})$$

$$\nabla_{\boldsymbol{x}}^2 g(\boldsymbol{x}) \preceq L$$

Minimize $\hat{\phi}(\boldsymbol{x}, \boldsymbol{x}_k)$ instead of minimizing $\phi(\boldsymbol{x})$ at $(k+1)$th iteration:

$$Prox_{\frac{1}{L}h}(\boldsymbol{v}_k) = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{x} - \boldsymbol{v}_k\|_2^2 + \frac{1}{L} h(\boldsymbol{x}) : \text{Closed-Form}$$

$$\boldsymbol{v}_k = \boldsymbol{x}_k - \frac{1}{L} \nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k)$$

[1] Euclidean distance here. Bregman distance in general, [Bec17].

Acceleration – Nesterov/FISTA

Set $\boldsymbol{y}_1 = \boldsymbol{x}_0$ and $t_1 = 1$ and repeat the following at step $k \geq 1$

- $\boldsymbol{x}_k = Prox_{\frac{1}{L}h}(\boldsymbol{y}_k - \frac{1}{L}\nabla_{\boldsymbol{x}} g(\boldsymbol{y}_k))$
- $\boxed{t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}}$
- $\boldsymbol{y}_{k+1} = \boldsymbol{x}_k + \boxed{\frac{t_k - 1}{t_{k+1}}}(\boldsymbol{x}_k - \boldsymbol{x}_{k-1})$

Acceleration – Nesterov/FISTA

Set $\boldsymbol{y}_1 = \boldsymbol{x}_0$ and $t_1 = 1$ and repeat the following at step $k \geq 1$

- $\boldsymbol{x}_k = Prox_{\frac{1}{L}h}(\boldsymbol{y}_k - \frac{1}{L}\nabla_{\boldsymbol{x}} g(\boldsymbol{y}_k))$
- $\boxed{t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}}$
- $\boldsymbol{y}_{k+1} = \boldsymbol{x}_k + \boxed{\frac{t_k - 1}{t_{k+1}}}(\boldsymbol{x}_k - \boldsymbol{x}_{k-1})$

Convergence Speed:

Proximal: $\mathcal{O}(\frac{1}{k})$

Acceleration: $\mathcal{O}(\frac{1}{k^2})$

Acceleration – Nesterov/FISTA

Set $\boldsymbol{y}_1 = \boldsymbol{x}_0$ and $t_1 = 1$ and repeat the following at step $k \geq 1$

- $\boldsymbol{x}_k = Prox_{\frac{1}{L}h}(\boldsymbol{y}_k - \frac{1}{L}\nabla_{\boldsymbol{x}} g(\boldsymbol{y}_k))$
- $\boxed{t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}}$
- $\boldsymbol{y}_{k+1} = \boldsymbol{x}_k + \boxed{\frac{t_k - 1}{t_{k+1}}}(\boldsymbol{x}_k - \boldsymbol{x}_{k-1})$

Convergence Speed:

Proximal: $\mathcal{O}(\frac{1}{k})$

Acceleration: $\mathcal{O}(\frac{1}{k^2})$

Can we do better?

No closed-form —— $Prox_{\frac{1}{L}h}(\cdot)$ —— RED

Weighted Proximal Methods

Linearizing $g(\boldsymbol{x})$ with higher accuracy:

$$g(\boldsymbol{x}) + h(\boldsymbol{x}) \leq \hat{\phi}(\boldsymbol{x}, \boldsymbol{x}_k)$$

$$\hat{\phi}(\boldsymbol{x}, \boldsymbol{x}_k) \triangleq g(\boldsymbol{x}_k) + < \nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k), \boldsymbol{x} - \boldsymbol{x}_k > + \frac{1}{2a_k} \boxed{(\boldsymbol{x} - \boldsymbol{x}_k)^{\mathcal{T}} \boldsymbol{B}_k (\boldsymbol{x} - \boldsymbol{x}_k)} + h(\boldsymbol{x})$$

a_k stepsize or use 1 and $\boldsymbol{B}_k \succ 0$. Define

$$Prox_{a_k h}^{WPM}(\boldsymbol{v}_k) = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{x} - \boldsymbol{v}_k\|_{\boldsymbol{B}_k}^2 + a_k h(\boldsymbol{x}) \text{ : No Closed-Form}$$

$$\boldsymbol{v}_k = \boldsymbol{x}_k - a_k \boldsymbol{B}_k^{-1} \nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k)$$

Weighted Proximal Methods

Linearizing $g(\boldsymbol{x})$ with higher accuracy:

$$g(\boldsymbol{x}) + h(\boldsymbol{x}) \leq \hat{\phi}(\boldsymbol{x}, \boldsymbol{x}_k)$$

$$\hat{\phi}(\boldsymbol{x}, \boldsymbol{x}_k) \triangleq g(\boldsymbol{x}_k) + < \nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k), \boldsymbol{x} - \boldsymbol{x}_k > + \frac{1}{2a_k} \boxed{(\boldsymbol{x} - \boldsymbol{x}_k)^{\mathcal{T}} \boldsymbol{B}_k (\boldsymbol{x} - \boldsymbol{x}_k)} + h(\boldsymbol{x})$$

a_k stepsize or use 1 and $\boldsymbol{B}_k \succ 0$. Define

$$Prox_{a_k h}^{WPM}(\boldsymbol{v}_k) = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{x} - \boldsymbol{v}_k\|_{\boldsymbol{B}_k}^2 + a_k h(\boldsymbol{x}) : \text{No Closed-Form}$$

$$\boldsymbol{v}_k = \boldsymbol{x}_k - a_k \boldsymbol{B}_k^{-1} \nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k)$$

In RED:

Remind the denoising $\boldsymbol{f}(\boldsymbol{x})$ in RED: high complexity

Weighted Proximal Methods

Linearizing $g(\boldsymbol{x})$ with higher accuracy:

$$g(\boldsymbol{x}) + h(\boldsymbol{x}) \leq \hat{\phi}(\boldsymbol{x}, \boldsymbol{x}_k)$$

$$\hat{\phi}(\boldsymbol{x}, \boldsymbol{x}_k) \triangleq g(\boldsymbol{x}_k) + < \nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k), \boldsymbol{x} - \boldsymbol{x}_k > + \frac{1}{2a_k} \boxed{(\boldsymbol{x} - \boldsymbol{x}_k)^{\mathcal{T}} \boldsymbol{B}_k (\boldsymbol{x} - \boldsymbol{x}_k)} + h(\boldsymbol{x})$$

a_k stepsize or use 1 and $\boldsymbol{B}_k \succ 0$. Define

$$Prox_{a_k h}^{WPM}(\boldsymbol{v}_k) = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{x} - \boldsymbol{v}_k\|_{\boldsymbol{B}_k}^2 + a_k h(\boldsymbol{x}) : \text{No Closed-Form}$$

$$\boldsymbol{v}_k = \boldsymbol{x}_k - a_k \boldsymbol{B}_k^{-1} \nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k)$$

In RED:

Remind the denoising $\boldsymbol{f}(\boldsymbol{x})$ in RED: high complexity

To reduce the calling of $\boldsymbol{f}(\boldsymbol{x})$, we set

$$\boxed{g(\boldsymbol{x}) = \lambda \rho(\boldsymbol{x})}$$

and

$$\boxed{h(\boldsymbol{x}) = \ell(\boldsymbol{x}, \boldsymbol{y})}$$

The Choice of $\boldsymbol{B}_k - \underbrace{\ell(\boldsymbol{x}, \boldsymbol{y})}_{h(\boldsymbol{x})} + \underbrace{\frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}\left(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})\right)}_{g(\boldsymbol{x})}$

The Choice of $\boldsymbol{B}_k$ – $\underbrace{\ell(\boldsymbol{x}, \boldsymbol{y})}_{h(\boldsymbol{x})} + \underbrace{\frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))}_{g(\boldsymbol{x})}$

Proximal method:

$$Prox_{\frac{1}{L}h}(\boldsymbol{v}_k) = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{v}_k\|_2^2 + \frac{1}{L}h(\boldsymbol{x}) : \text{Closed-Form}$$

$$\boldsymbol{v}_k = \boldsymbol{x}_k - \frac{1}{L}\nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k)$$

WPMs:

$$Prox^{WPM}_{a_k h}(\boldsymbol{v}_k) = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{v}_k\|^2_{\boldsymbol{B}_k} + a_k h(\boldsymbol{x}) : \text{No Closed-Form}$$

$$\boldsymbol{v}_k = \boldsymbol{x}_k - a_k \boldsymbol{B}_k^{-1}\nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k)$$

- $\boldsymbol{B}_k = \lambda \boldsymbol{I}$ and $a_k = 1$: recover proximal method

The Choice of $\boldsymbol{B}_k$ – $\underbrace{\ell(\boldsymbol{x}, \boldsymbol{y})}_{h(\boldsymbol{x})} + \underbrace{\frac{\lambda}{2}\boldsymbol{x}^{\mathcal{T}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))}_{g(\boldsymbol{x})}$

Proximal method:

$$Prox_{\frac{1}{L}h}(\boldsymbol{v}_k) = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{v}_k\|_2^2 + \frac{1}{L}h(\boldsymbol{x}) : \text{Closed-Form}$$

$$\boldsymbol{v}_k = \boldsymbol{x}_k - \frac{1}{L}\nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k)$$

WPMs:

$$Prox_{a_k h}^{WPM}(\boldsymbol{v}_k) = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{v}_k\|_{\boldsymbol{B}_k}^2 + a_k h(\boldsymbol{x}) : \text{No Closed-Form}$$

$$\boldsymbol{v}_k = \boldsymbol{x}_k - a_k \boldsymbol{B}_k^{-1}\nabla_{\boldsymbol{x}} g(\boldsymbol{x}_k)$$

- $\boldsymbol{B}_k = \lambda \boldsymbol{I}$ and $a_k = 1$: recover proximal method
- $\boldsymbol{B}_k$: the Hessian of $g(\boldsymbol{x}) \rightarrow$ Quasi-Newton [NW06]
- ...

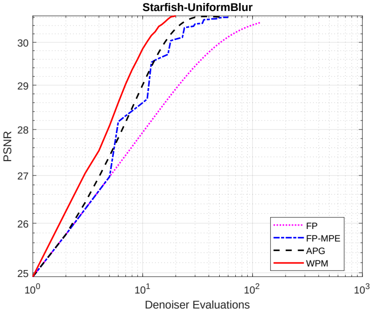
Estimate $\boldsymbol{B}_k$ — SR1

Algorithm 1 SR1

Initialization: $k = 1$, $\gamma = 1.25$, $\delta = 10^{-8}$, $\boldsymbol{x}_k$, $\boldsymbol{x}_{k-1}$, $\nabla g(\boldsymbol{x}_k)$, $\nabla g(\boldsymbol{x}_{k-1})$.

if $k = 1$ **then**
  $\boldsymbol{B}_k \leftarrow \alpha \boldsymbol{I}$
else
  Set $\boxed{\boldsymbol{s}_k} \leftarrow \boldsymbol{x}_k - \boldsymbol{x}_{k-1}$ and $\boxed{\boldsymbol{m}_k} \leftarrow \nabla g(\boldsymbol{x}_k) - \nabla g(\boldsymbol{x}_{k-1})$
  Calculate $\tau \leftarrow \gamma \frac{\|\boldsymbol{m}_k\|_2^2}{\langle \boldsymbol{s}_k, \boldsymbol{m}_k \rangle}$
  if $\tau < 0$ **then**
    $\boxed{\boldsymbol{B}_k \leftarrow \alpha \boldsymbol{I}}$
  else
    $\boxed{\boldsymbol{H}_0} \leftarrow \tau \boldsymbol{I}$
    if $|\langle \boldsymbol{m}_k - \boldsymbol{H}_0 \boldsymbol{s}_k, \boldsymbol{s}_k \rangle| \leq \delta \|\boldsymbol{s}_k\|_2 \|\boldsymbol{m}_k - \boldsymbol{H}_0 \boldsymbol{s}_k\|_2$ **then**
      $\boldsymbol{u}_k \leftarrow 0$
    else
      $\boldsymbol{u}_k \leftarrow \frac{\boldsymbol{m}_k - \boldsymbol{H}_0 \boldsymbol{s}_k}{\sqrt{\langle \boldsymbol{m}_k - \boldsymbol{H}_0 \boldsymbol{s}_k, \boldsymbol{s}_k \rangle}}$
    end if
    $\boxed{\boldsymbol{B}_k \leftarrow \boldsymbol{H}_0 + \boldsymbol{u}_k \boldsymbol{u}_k^{\mathcal{T}}}$
  end if
end if
Return: $\boldsymbol{B}_k$

Image Deblurring - Uniform



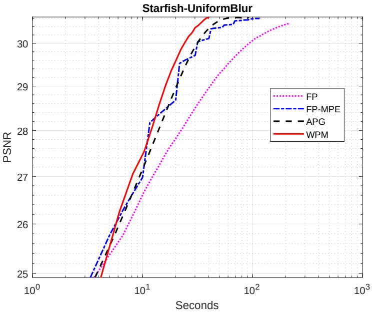
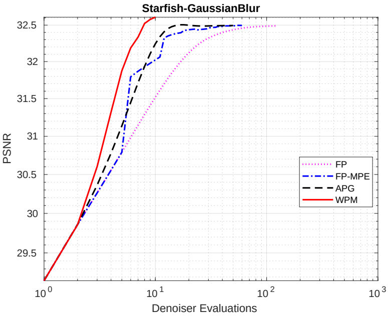


Image Deblurring - Gaussian



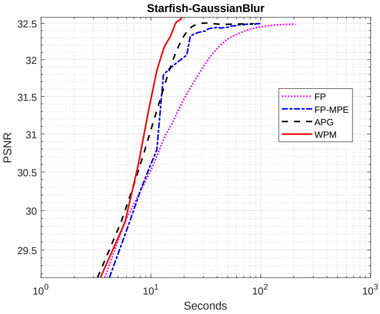
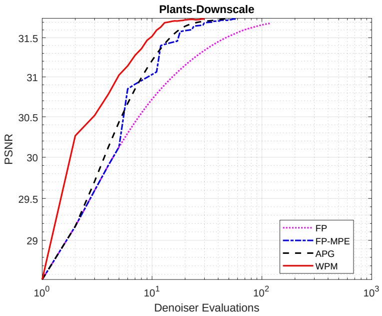
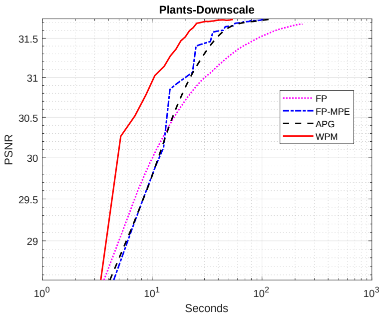


Image Super-Resolution (SR)





More Results [HYZ19]

FP – 200 denoiser evaluations

Denoiser evaluations, other methods – compariable PSNR

1st and 2nd row: deblurring with uniform and Gaussian blurs.

3rd row: SR

	FP-MPE	APG	WPM
Butterfly	54	34	25
	54	26	17
	80	51	26
Boats	24	20	21
	60	34	22
	36	20	12
House	24	18	19
	62	26	25
	18	15	10
Parrot	39	30	20
	52	40	36
	49	31	28
Lena	48	34	29
	47	16	15
	37	26	18
Barbara	14	12	11
	48	23	16
	17	15	11
Peppers	42	29	22
	41	40	34
	38	30	28
Leaves	50	41	34
	36	18	14
	60	41	12

Conclusion

WPMs are good if no closed-form solution exists for the proximal operator.

Thanks & Questions?

Amir Beck.
First-Order Methods in Optimization, volume 25.
SIAM, 2017.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al.
Distributed optimization and statistical learning via the alternating direction method of multipliers.
Foundations and Trends® in Machine Learning, 3(1):1–122, 2011.

Tao Hong, Yaniv Romano, and Michael Elad.
Acceleration of red via vector extrapolation.
Journal of Visual Communication and Image Representation, page 102575, 2019.

Tao Hong, Irad Yavneh, and Michael Zibulevsky.
Solving red via weighted proximal methods.
arXiv preprint arXiv:1905.13052, 2019.

Jorge Nocedal and Stephen J. Wright.
Numerical Optimization.

Springer, 2006.

Yaniv Romano, Michael Elad, and Peyman Milanfar.
The little engine that could: Regularization by denoising (red).
SIAM Journal on Imaging Sciences, 10(4):1804–1844, 2017.

Edward T Reehorst and Philip Schniter.
Regularization by denoising: Clarifications and new interpretations.
IEEE Transactions on Computational Imaging, 5(1):52–67, 2019.