Variational Methods for Image Restoration

Yuanfeng Shi, 1800010697

October 12, 2019

1 Finite Difference Approximation of Derivatives

Difference among different discretizations (h=1)

$$\Delta v|_{i,j} \simeq \lambda \frac{v_{i+1,j}+v_{i-1,j}+v_{i,j+1}+v_{i,j-1}-4v_{i,j}}{h^2} + (1-\lambda)\frac{v_{i+1,j+1}+v_{i-1,j+1}+v_{i+1,j-1}+v_{i-1,j-1}-4v_{i,j}}{2h^2}$$

The second approximation of derivatives can be seen as the 45 degree rotation of the first approximation, and its step size is $\sqrt{2}h$. Consider

$$(1-\lambda)\frac{v_{i+1,j+1}+v_{i-1,j+1}+v_{i+1,j-1}+v_{i-1,j-1}-4v_{i,j}}{(\sqrt{2}h)^2}$$

In order to make the norms of different discrete gradients are equal, $2*(1-\lambda)=\lambda, \lambda=\frac{2}{3}$.

As for the first order,

$$\frac{\partial v}{\partial x}|_{i,j} \simeq \lambda \frac{v_{i+1,j}-v_{i-1,j}}{2h} + \frac{(1-\lambda)}{2}(\frac{v_{i+1,j+1}-v_{i-1,j+1}}{2h} + \frac{v_{i+1,j-1}-v_{i-1,j-1}}{2h})$$

$\sqrt{2}*(1-\lambda)=\lambda, \lambda = 2-\sqrt{2}$.

2 Introduction

Image restoration, including image denoising, deblurring, superresolution, *etc.*, is one of the most important areas in imaging science. Its major purpose is to obtain high quality reconstructions of images corrupted in various ways during imaging, acquisiting and storing. The basic linear restoration model is usually characterized as solving the inverse problem

$$f = Au + \eta \tag{1}$$

The fundamental problem in image reconstruction is to design an appropriate regularization to characterize natural images. Rudin and Osher introduced a regularization term, and the functional is like:

$$E_{ROF}(u) := \|f-Au\|_{L^2(\Omega)} + |u|_{TV(\Omega)}. \tag{2}$$

Here the Total Variation (TV) seminorm is a generalization of the integration of the derivative of u: $\int_\Omega |u'|$. The ROF model can preserve the edge while deblur or denoise in the subdomain successfully.

To solve the ROF method, we use the Split Bergman method.

3 Split Bregman Method

The Split Bregman method is a technique for solving a variety of L1-regularized optimization problems, and is particularly effective for problems involving total-variation regularization. Split Bregman is one of the fastest solvers for Total-Variation denoising, image reconstruction from Fourier coefficients, convex image segmentation, and many other problems. The method is a re-interpretation of the alternating direction method of multipliers that is specially adapted to L1 problems.

Let us consider a general case:

$$\min \|\lambda \cdot Wu\|_1 + H(u), \tag{3}$$

where W is a $n \times n$ matrix and H is convex and differentiable. We begin with a simple substitution:

$$\min_{u,d} \|\lambda \cdot d\|_1 + H(u) \quad s.t. \quad d = Wu.$$

Recall the Augmented Lagrangian:

$$\mathcal{L}_\mu(u,d,p) = \|\lambda \cdot d\|_1 + H(u) + p^T(d - Mu) + \frac{\mu}{2}\|Wu - d\|_2^2.$$

Solving the saddle point problem by:

$$\begin{cases} (u_{k+1}, d_{k+1}) & = argmin_{u,d}\|\lambda \cdot d\|_1 + H(u) + p_k^T(d - Mu) + \frac{\mu}{2}\|Wu - d\|_2^2 \\ p_{k+1} & = p_k + \delta\mu(d_{k+1} - Wu_{k+1}) \end{cases}$$

Which is equivalent to the following subroutine:

$$\begin{cases} (u_{k+1}, d_{k+1}) & = argmin_{u,d}\|\lambda \cdot d\|_1 + H(u) + \frac{\mu}{2}\|Mu - d + b_k\|_2^2 \\ b_{k+1} & = b_k + \delta(Wu_{k+1} - d_{k+1}) \end{cases}$$

4 Detailed Algorithm

We use the discretization in Section 1. Assume u is a function defined on $[0,1]^2$.

Define the space with grid size $h := \frac{1}{n}$:

$$\Omega_n := \{(\frac{i}{n}, \frac{j}{n}) \in [0,1]^2 : i,j \in \mathbb{Z}\} \tag{4}$$

equipped with the following norm :

$$\|\boldsymbol{v}\|_{p,n} := \big(\sum_{i=0,j=0}^{n} |\boldsymbol{v}(i,j)|^p\big)^{1/p} \tag{5}$$

Then the discrete functional is proposed as:

$$E_n(\boldsymbol{u}) = \frac{1}{2}\|\boldsymbol{A}\boldsymbol{u} - \boldsymbol{f}\|_{2,n}^2 + \frac{\lambda}{2}\big[\|\frac{\partial}{\partial x}\boldsymbol{u}\|_{1,n} + \|\frac{\partial}{\partial y}\boldsymbol{u}\|_{1,n}\big]. \tag{6}$$

And it can be approximately realized by a $2n \times n$ matrix W:

$$E_n(\boldsymbol{u}) = \frac{1}{2}\|\boldsymbol{A}\boldsymbol{u} - \boldsymbol{f}\|_{2,n}^2 + \frac{\lambda}{2}\|\boldsymbol{W}\boldsymbol{u}\|_{1,n}. \tag{7}$$

To solve (7), we adopt the Split Bregman Algorithm introduced in Section 3. For simplicity, we drop the subscript n when the grid size is fixed More explicitly, let $H(\boldsymbol{u}) = \frac{1}{2}\|\boldsymbol{Au} - \boldsymbol{f}\|_2^2$, we get

$$\begin{cases} \boldsymbol{u}_{k+1} & = argmin_{\boldsymbol{u}} \frac{1}{2}\|\boldsymbol{Au} - \boldsymbol{f}\|_2^2 + \frac{\mu}{2}\|\boldsymbol{Wu} - \boldsymbol{d}_k + \boldsymbol{b}_k\|_2^2 \\ \boldsymbol{d}_{k+1} & = argmin_{\boldsymbol{d}} \lambda\|\boldsymbol{d}\|_1 + \frac{\mu}{2}\|\boldsymbol{d} - \boldsymbol{W}\boldsymbol{u}_{k+1} - \boldsymbol{b}_k\|_2^2 \\ \boldsymbol{b}_{k+1} & = \boldsymbol{b}_k + \delta(\boldsymbol{W}\boldsymbol{u}_{k+1} - \boldsymbol{d}_{k+1}) \end{cases}$$

Each step in the subroutine has an explicit solution, so we obtain the efficient algorithm:

Algorithm 1 Split Bregman For Analysis Approach

```
1: set $\boldsymbol{d}_0$, $\boldsymbol{b}_0$ as zero.
2: $\boldsymbol{u} = \boldsymbol{f}$
3: repeat
4:     $\boldsymbol{u}_{k+1} = \left[\boldsymbol{A}^T\boldsymbol{A} + \mu\boldsymbol{W}^T\boldsymbol{W}\right]^{-1}[\boldsymbol{A}^T\boldsymbol{f} + \mu\boldsymbol{W}^T(\boldsymbol{d}_k - \boldsymbol{b}_k)]$
5:     $\boldsymbol{d}_{k+1} = \mathcal{T}_{\lambda/\mu}(\boldsymbol{W}\boldsymbol{u}_{k+1} + \boldsymbol{b}_k)$
6:     $\boldsymbol{b}_{k+1} = \boldsymbol{b}_k + \delta(\boldsymbol{W}\boldsymbol{u}_{k+1} - \boldsymbol{d}_{k+1})$
7: until $\frac{\|\boldsymbol{W}\boldsymbol{u}_{k+1} - \boldsymbol{d}_{k+1}\|}{\|f\|_2} < tol$ or achieve maximal iter
```

Remark 1. *1. $\mathcal{T}_\alpha$ is the soft thresholding function $x \mapsto \frac{x}{|x|}\max(x - \alpha, 0)$*

2. In general, we use the sparse matrix to lower the computitional complexity.

3. We use the function gmres for sparse matrix to solve the calculation of $\boldsymbol{u}$.

4. The more details of the algorithm is shown in the MATLAB code.

5 Numerical Result

In this section, we conduct some numerical simulations on image deblurring.We test the algorithm in different blur settings and noise settings. The convolution operator is generated by “fspecial” function in MATLAB and the noise is generated by “randn” function. For the quantitative comparison, we calculate the peak signal to noise ratio (PSNR) value and the SSIM value, and the results are below.

noise	index	peppers256			barbara		
		Origin	End	Max	Origin	End	Max
0.01	PSNR	24.79	29.80	30.02	24.98	28.17	28.24
	SSIM	0.8641	0.8895	0.8911	0.7559	0.8221	0.8431
0.01	PSNR	24.53	27.97	28.12	24.51	26.54	26.68
	SSIM	0.7566	0.7977	0.8031	0.6752	0.7304	0.7607

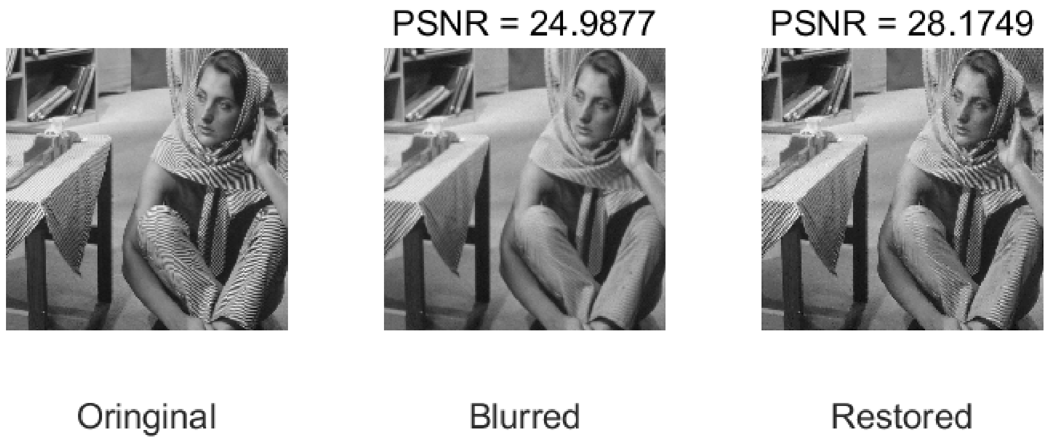

Figure 1: Image 'barbara': kernel = fspecial(15,1) noise = 0.01*randn(n*n)

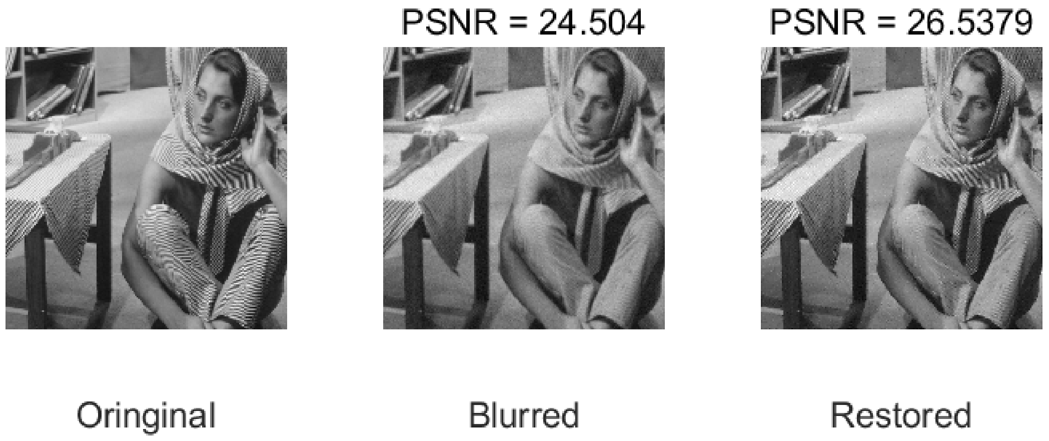

Figure 2: Image 'barbara': kernel = fspecial(15,1) noise = 0.02*randn(n*n)

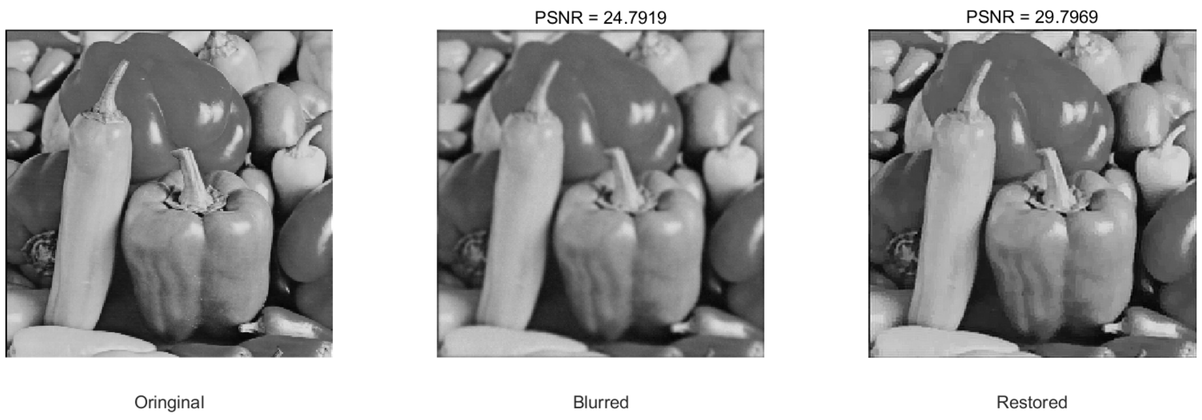

Figure 3: Image 'peppers256': kernel = fspecial(15,1) noise = 0.01*randn(n*n)

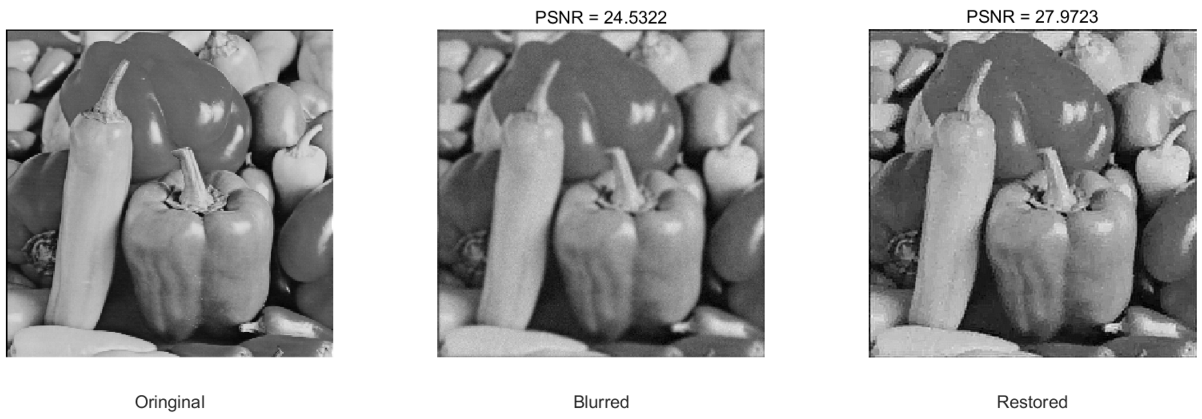

Figure 4: Image 'peppers256': kernel = fspecial(15,1) noise = 0.02*randn(n*n)