MA3111: Mathematical Image Processing
Variational Image Deblurring

Suh-Yuh Yang (楊肅煜)

Department of Mathematics, National Central University
Jhongli District, Taoyuan City 320317, Taiwan

First version: October 23, 2016/Last updated: August 25, 2024

Outline of "variational image deblurring"

In this lecture, we will give a brief introduction to the topics:

- *The blurring kernels of motion blur and Gaussian blur.*
- *The standard total variation model for variational image deblurring.*

The material of this lecture is mainly based on

- T. F. Chan and C.-K. Wong, Total variation blind deconvolution, *IEEE Transaction on Image Processing,* 7 (1998), pp. 370-375.
- Y. Wang, W. Yin, and Y. Zhang, A fast algorithm for image deblurring with total variation regularization, *CAAM Technical Report TR 07-10,* 2007, Rice University.

Blurry and noisy image restoration

- **Image restoration (影像修復):** One of the important tasks in image processing is to recover images from noisy and blurry observations.

  To recover a sharp image from its blurry observation is the problem known as image deblurring (影像去模糊).

- These blurring artifacts may come from different sources, such as atmospheric turbulence, diffraction, optical defocusing, camera shaking, and more.

- The blurry and noisy observation is generally modeled as

$$f(\boldsymbol{x}) = (K\bar{u})(\boldsymbol{x}) + n(\boldsymbol{x}), \quad \boldsymbol{x} \in \overline{\Omega},$$

  where $\bar{u}$ is the clean image, n is the Gaussian noise, and K is a blurring operator.

  We may assume the image domain is $\overline{\Omega}$ and zero-valued for all $\boldsymbol{x} \in \mathbb{R}^2 \setminus \overline{\Omega}$.

Linear and shift-invariant blurring operator K

The blurring operator K is typically assumed to be a *"linear" and "shift-invariant"* operator, expressed in the convolutional form:

$$(Ku)(\boldsymbol{x}) = \int_\Omega h(\boldsymbol{x}-\boldsymbol{s})u(\boldsymbol{s})d\boldsymbol{s} =: (h \star u)(\boldsymbol{x}), \quad \boldsymbol{x} \in \overline{\Omega},$$

where $\star$ denotes the convolution operation and h is the so-called point spread function (blurring kernel) associated with the linear blurring operator K. Therefore, the image deblurring is also called the image deconvolution.

- *K is linear:*

$$\begin{aligned}\big(K(\alpha u+\beta v)\big)(\boldsymbol{x}) &= \int_\Omega h(\boldsymbol{x}-\boldsymbol{s})\big(\alpha u(\boldsymbol{s})+\beta v(\boldsymbol{s})\big)d\boldsymbol{s} \\ &= \cdots = \alpha(Ku)(\boldsymbol{x})+\beta(Kv)(\boldsymbol{x}), \quad \forall\, \boldsymbol{x}\in\overline{\Omega}.\end{aligned}$$

- *K is shift-invariant:* Let $g(\boldsymbol{x}) = f(\boldsymbol{x}-\boldsymbol{\tau})$ for $\boldsymbol{\tau} \in \mathbb{R}^2$. Then

$$\begin{aligned}(Kg)(\boldsymbol{x}) &= \int_{\mathbb{R}^2} h(\boldsymbol{x}-\boldsymbol{s})g(\boldsymbol{s})d\boldsymbol{s} = (h\star g)(\boldsymbol{x}) = (g\star h)(\boldsymbol{x}) \\ &= \int_{\mathbb{R}^2} g(\boldsymbol{x}-\boldsymbol{s})h(\boldsymbol{s})d\boldsymbol{s} = \int_{\mathbb{R}^2} f(\boldsymbol{x}-\boldsymbol{\tau}-\boldsymbol{s})h(\boldsymbol{s})d\boldsymbol{s} \\ &= (f\star h)(\boldsymbol{x}-\boldsymbol{\tau}) = (Kf)(\boldsymbol{x}-\boldsymbol{\tau}), \quad \forall\, \boldsymbol{x}\in\overline{\Omega}.\end{aligned}$$

Creating a 2-D blurring filter `H` in Matlab

Motion blur:

```
>> H = fspecial('motion', len, theta)
```

returns a filter to approximate the linear motion of a camera by the length of `len` pixels of the motion, with an angle of `theta` degrees in a counterclockwise direction.

The default `len` *is* 9 *pixels and the default* `theta` *is* 0 *degree.*

Examples:

```
>> H = fspecial('motion', 5, 45)
```

$$H = \begin{bmatrix} 0 & 0 & 0 & 0.0501 & 0.0304 \\ 0 & 0 & 0.0519 & 0.1771 & 0.0501 \\ 0 & 0.0519 & 0.1771 & 0.0519 & 0 \\ 0.0501 & 0.1771 & 0.0519 & 0 & 0 \\ 0.0304 & 0.0501 & 0 & 0 & 0 \end{bmatrix}$$

Motion blur (cont'd)

```
>> H = fspecial('motion', 5, 30)
```

$$H = \begin{bmatrix} 0 & 0 & 0.0268 & 0.1268 & 0.1464 \\ 0 & 0.1000 & 0.2000 & 0.1000 & 0 \\ 0.1464 & 0.1268 & 0.0268 & 0 & 0 \end{bmatrix}$$

```
>> H = fspecial('motion', 5, 60)
```

$$H = \begin{bmatrix} 0 & 0 & 0.1464 \\ 0 & 0.1000 & 0.1268 \\ 0.0268 & 0.2000 & 0.0268 \\ 0.1268 & 0.1000 & 0 \\ 0.1464 & 0 & 0 \end{bmatrix}$$

A motion filter and blurred image: cameraman

Read image `cameraman.png` and display it:

```
>> I = imread('cameraman.png');
>> imshow(I);
```

Create a motion filter and use it to blur the image:

```
>> H = fspecial('motion', 30, 45);
>> motion_blur = imfilter(I, H, 'replicate');
```

Display the blurred image:

```
>> imshow(motion_blur);
```

Gaussian blur

```
>> H = fspecial('gaussian', hsize, sigma)
```

returns a rotationally symmetric Gaussian lowpass filter of size `hsize` with standard deviation `sigma`.

Example:

```
>> H = fspecial('gaussian', 5, 1)
```

$$H = \begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix}$$

Here `fspecial` creates Gaussian filters using

$$H_g(n_1, n_2) := e^{\dfrac{-(n_1^2 + n_2^2)}{2\sigma^2}} \quad \text{and} \quad H(n_1, n_2) := \frac{H_g(n_1, n_2)}{\sum_{n_1} \sum_{n_2} H_g}.$$

A Gaussian filter and blurred image: cameraman

Read image `cameraman.png` and display it:

```
>> I = imread('cameraman.png');
>> imshow(I);
```

Create a Gaussian filter and use it to blur the image:

```
>> H = fspecial('gaussian', 30, 5);
>> gaussian_blur = imfilter(I, H, 'replicate');
```

Display the blurred image:

```
>> imshow(gaussian_blur);
```

Blurry and noisy image restoration

The total variation (TV) regularization has become one of the standard techniques known for preserving sharp discontinuities such as edges and object boundaries.

Let $f : \overline{\Omega} \subset \mathbb{R}^2 \to \mathbb{R}$ be a given blurry and noisy image in $L^2(\Omega)$. The standard total variation model recovers an image from f by solving the TV/L2 problem:

$$\min_u \int_\Omega |\nabla u(\boldsymbol{x})| d\boldsymbol{x} + \frac{\lambda}{2} \int_\Omega \left((Ku)(\boldsymbol{x}) - f(\boldsymbol{x})\right)^2 d\boldsymbol{x},$$

where $\lambda > 0$ is a model parameter, K is a linear blurring operator, u is the unknown image to be restored, and

$$|\nabla u(\boldsymbol{x})| := \|\nabla u(\boldsymbol{x})\|_2 = \sqrt{(\partial u/\partial x)^2 + (\partial u/\partial y)^2}.$$

Here, we assume that $(Ku)(\boldsymbol{x}) = (h \star u)(\boldsymbol{x})$ for all $\boldsymbol{x} \in \overline{\Omega}$ and the point spread function h is given.

If both the blur kernel h and the latent sharp image u are unknown, the problem is called "blind image deblurring" or "blind image deconvolution."

The energy functional

Since the energy functional in the TV/L2 problem is convex, u is optimal if and only if it satisfies the first-order optimality condition. Define the energy functional

$$E[u] := \int_\Omega |\nabla u(\boldsymbol{x})| + \frac{\lambda}{2}\big((Ku)(\boldsymbol{x}) - f(\boldsymbol{x})\big)^2 d\boldsymbol{x}.$$

For any smooth function η with $\eta = 0$ on $\partial\Omega$, let $\Phi(\varepsilon) := E[u + \varepsilon\eta]$, then we have

$$\begin{aligned}
0 &= \Phi'(0) = \frac{d}{d\varepsilon}\Phi(\varepsilon)\Big|_{\varepsilon=0} = \lim_{\varepsilon\to 0}\frac{E[u+\varepsilon\eta] - E[u]}{\varepsilon - 0} \\
&= \lim_{\varepsilon\to 0}\frac{1}{\varepsilon}\Big(\int_\Omega |\nabla u(\boldsymbol{x}) + \varepsilon\nabla\eta(\boldsymbol{x})| + \frac{\lambda}{2}\big((Ku + \varepsilon K\eta)(\boldsymbol{x}) - f(\boldsymbol{x})\big)^2 d\boldsymbol{x} \\
&\qquad - \int_\Omega |\nabla u(\boldsymbol{x})| + \frac{\lambda}{2}\big((Ku)(\boldsymbol{x}) - f(\boldsymbol{x})\big)^2 d\boldsymbol{x}\Big) \\
&= \Big(\int_\Omega \frac{\nabla u(\boldsymbol{x}) + \varepsilon\nabla\eta(\boldsymbol{x})}{|\nabla u(\boldsymbol{x}) + \varepsilon\nabla\eta(\boldsymbol{x})|}\Big|_{\varepsilon=0} \cdot \nabla\eta(\boldsymbol{x}) d\boldsymbol{x}\Big) \\
&\quad + \lim_{\varepsilon\to 0}\frac{1}{\varepsilon}\frac{\lambda}{2}\Big(\int_\Omega \big(\varepsilon(K\eta)(\boldsymbol{x})\big)^2 + 2\varepsilon(K\eta)(\boldsymbol{x})\big((Ku)(\boldsymbol{x}) - f(\boldsymbol{x})\big)\Big).
\end{aligned}$$

The Euler-Lagrange equation

Then, by Green's formula, we obtain

$$\begin{aligned} 0 &= \int_\Omega -\nabla \cdot \frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|}\eta(\boldsymbol{x}) + \lambda (K\eta)(\boldsymbol{x})\big((Ku)(\boldsymbol{x}) - f(\boldsymbol{x})\big) d\boldsymbol{x} \\ &= \int_\Omega -\nabla \cdot \frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|}\eta(\boldsymbol{x}) + \lambda \eta(\boldsymbol{x}) K^*\big((Ku)(\boldsymbol{x}) - f(\boldsymbol{x})\big) d\boldsymbol{x} \\ &= \int_\Omega \left(-\nabla \cdot \frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|} + \lambda K^*\big((Ku)(\boldsymbol{x}) - f(\boldsymbol{x})\big)\right)\eta(\boldsymbol{x}) d\boldsymbol{x} \end{aligned}$$

for any smooth function η with $\eta = 0$ on $\partial\Omega$, where K^* is the adjoint operator of K. Therefore, we attain the Euler-Lagrange equation,

$$-\nabla \cdot \left(\frac{\nabla u}{|\nabla u|}\right) + \lambda K^*(Ku - f) = 0 \quad \text{for } \boldsymbol{x} \in \Omega,$$

or equivalently,

$$\nabla \cdot \left(\frac{\nabla u}{|\nabla u|}\right) - \lambda K^*(Ku - f) = 0 \quad \text{for } \boldsymbol{x} \in \Omega,$$

along with the Neumann boundary condition, $\partial u(\boldsymbol{x})/\partial \boldsymbol{n} = 0$ on $\partial\Omega$.

The adjoint operator

Let $\mathcal{V}$ be a real (or complex) Hilbert space with inner product $\langle \cdot, \cdot \rangle$, e.g., $L^2(\Omega)$ with the inner product $\langle f, g \rangle := \int_\Omega fg \, d\Omega$.

- Consider a continuous (i.e., bounded) linear operator $T : \mathcal{V} \to \mathcal{V}$. Then the adjoint of T is the continuous linear operator $T^* : \mathcal{V} \to \mathcal{V}$ satisfying

$$\langle Tx, y \rangle = \langle x, T^* y \rangle, \quad \forall\, x, y \in \mathcal{V}.$$

- Existence and uniqueness of this operator follows from the Riesz representation theorem.
- This can be seen as a generalization of the adjoint matrix of a square matrix , i.e., the conjugate transpose of a square matrix. For example, let $A \in \mathbb{R}^{3\times 3}$. Then

$$\langle A\boldsymbol{x}, \boldsymbol{y} \rangle = \boldsymbol{y}^\top A\boldsymbol{x} = \langle \boldsymbol{x}, A^\top \boldsymbol{y} \rangle, \quad \forall\, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3.$$

What is the adjoint operator K^* of K?

Suppose that the linear and shift-invariant blurring operator $K : L^2(\Omega) \to L^2(\Omega)$ is defined as

$$(Ku)(\boldsymbol{x}) := (h \star u)(\boldsymbol{x}) = \int_\Omega h(\boldsymbol{x} - \boldsymbol{s}) u(\boldsymbol{s}) d\boldsymbol{s} \quad \forall\, \boldsymbol{x} \in \overline{\Omega},$$

where h is the given kernel function.

$$\begin{aligned}\langle Ku, v\rangle_{L^2(\Omega)} &= \int_\Omega \left(\int_\Omega h(\boldsymbol{x} - \boldsymbol{s}) u(\boldsymbol{s}) d\boldsymbol{s} \right) v(\boldsymbol{x}) d\boldsymbol{x} \\ &= \int_\Omega u(\boldsymbol{s}) \left(\int_\Omega h(\boldsymbol{x} - \boldsymbol{s}) v(\boldsymbol{x}) d\boldsymbol{x} \right) d\boldsymbol{s}.\end{aligned}$$

Let $\widetilde{h}(\boldsymbol{x}) = h(-\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^2$. Then for all $u, v \in L^2(\Omega)$, we have

$$\begin{aligned}\langle u, K^* v\rangle_{L^2(\Omega)} &= \langle Ku, v\rangle_{L^2(\Omega)} = \int_\Omega u(\boldsymbol{s}) \left(\int_\Omega \widetilde{h}(\boldsymbol{s} - \boldsymbol{x}) v(\boldsymbol{x}) d\boldsymbol{x} \right) d\boldsymbol{s} \\ &= \langle u, \widetilde{h} \star v\rangle_{L^2(\Omega)}.\end{aligned}$$

Therefore, $(K^* v)(\boldsymbol{x}) = (\widetilde{h} \star v)(\boldsymbol{x})$.

Nonlinear PDE based image restoration

Consider the E-L equation with the homogeneous BC, $\frac{\partial u}{\partial n} = 0$ on $\partial\Omega$.

$$\nabla \cdot \left(\frac{\nabla u}{|\nabla u|_\delta} \right) - \lambda K^*(Ku - f) = 0,$$

where $|\cdot|_\delta := \sqrt{|\cdot|^2 + \delta^2}$, $0 < \delta \ll 1$, to avoid division by zero.

- Rudin-Osher (1994) used the artificial time marching method:

  $$u \leftarrow u + \Delta t \left\{ \nabla \cdot \left(\frac{\nabla u}{|\nabla u|_\delta} \right) - \lambda K^*(Ku - f) \right\}.$$

  This method is very easy to implement but converges slowly due to the nonlinearity of the diffusion operator.

- Vogel-Oman (1996) used *a lagged diffusivity procedure* to partially overcome this difficulty by solving the following equation for $u^{(n+1)}$ iteratively:

  $$\nabla \cdot \left(\frac{\nabla u^{(n+1)}}{|\nabla u^{(n)}|_\delta} \right) - \lambda K^*(Ku^{(n+1)} - f) = 0.$$

An equivalent constrained convex problem

By introducing a new variable $\boldsymbol{w}(\boldsymbol{x}) := \nabla u(\boldsymbol{x})$, we obtain an equivalent constrained convex minimization problem:

$$\min_{u,\boldsymbol{w}} \int_\Omega |\boldsymbol{w}(\boldsymbol{x})| d\boldsymbol{x} + \frac{\lambda}{2} \int_\Omega \big((Ku)(\boldsymbol{x}) - f(\boldsymbol{x})\big)^2 d\boldsymbol{x},$$

$$\text{subject to } \ \boldsymbol{w}(\boldsymbol{x}) = \nabla u(\boldsymbol{x}),\ \boldsymbol{x} \in \Omega.$$

Wang-Yin-Zhang (2007) considered the L^2-norm-square penalty formulation to obtain the unconstrained problem:

$$\min_{u,\boldsymbol{w}} \int_\Omega |\boldsymbol{w}(\boldsymbol{x})| d\boldsymbol{x} + \frac{\lambda}{2} \int_\Omega \big((Ku)(\boldsymbol{x}) - f(\boldsymbol{x})\big)^2 d\boldsymbol{x} + \frac{\beta}{2} \int_\Omega \big(\boldsymbol{w}(\boldsymbol{x}) - \nabla u(\boldsymbol{x})\big)^2 d\boldsymbol{x},$$

where $\beta > 0$ is a sufficiently large penalty parameter in order to approximate the solution of the original problem.

The discrete form of the unconstrained problem

Suppose that $f = [f_{ij}]$ is an $N \times N$ digital image. Let us consider the discrete form of the unconstrained problem:

$$\min_{u,\boldsymbol{w}} \sum_{i,j=1}^{N} \|\boldsymbol{w}_{ij}\| + \frac{\lambda}{2}\|Ku - f\|_F^2 + \frac{\beta}{2}\sum_{i,j=1}^{N} \|(\partial^+ u)_{ij} - \boldsymbol{w}_{ij}\|^2,$$

where K is the discrete convolution operator, $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^2$, i.e., $\|\cdot\| := \|\cdot\|_2$, and $\|\cdot\|_F$ is the Frobenius norm,

$$\boldsymbol{w}_{ij} = \begin{pmatrix} (w_1)_{ij} \\ (w_2)_{ij} \end{pmatrix} \in \mathbb{R}^2.$$

Moreover, ∂^+ denotes the forward finite difference operator,

$$(\partial^+ u)_{ij} = \begin{pmatrix} (\partial_1^+ u)_{ij} \\ (\partial_2^+ u)_{ij} \end{pmatrix} = \begin{pmatrix} u_{i+1,j} - u_{ij} \\ u_{i,j+1} - u_{ij} \end{pmatrix} \in \mathbb{R}^2.$$

An alternating method

We will solve the discrete problem by alternately minimizing the objective function with respect to $\boldsymbol{w}$ while fixing u, and vice versa.

$\boldsymbol{w}$-subproblem: For a fixed u, we solve

$$\min_{\boldsymbol{w}} \sum_{i,j=1}^{N} \left(\|\boldsymbol{w}_{ij}\| + \frac{\beta}{2} \|\boldsymbol{w}_{ij} - (\partial^+ u)_{ij}\|^2 \right),$$

which permits a closed-form solution

$$\boldsymbol{w}_{ij} = \max\left(\|(\partial^+ u)_{ij}\| - \frac{1}{\beta},\ 0 \right) \frac{(\partial^+ u)_{ij}}{\|(\partial^+ u)_{ij}\|}, \quad 1 \le i, j, \le N,$$

where we follow the convention that $0 \cdot (0/0) := 0$. The computation complexity is of order $O(N^2)$.

An alternating method (cont'd)

u**-subproblem:** For a fixed $\boldsymbol{w} = (w_1, w_2)^\top$, we solve the following problem with a special structure:

$$\min_u \frac{\lambda}{2}\|Ku - f\|_F^2 + \frac{\beta}{2}\|\partial_1^+ u - w_1\|_F^2 + \frac{\beta}{2}\|\partial_2^+ u - w_2\|_F^2,$$

where $Ku = H \star u$ with a given blurring filter H, $\partial_1^+ u = [(\partial_1^+ u)_{ij}]$, $w_1 = [(w_1)_{ij}]$, and so on, and all are matrices in $\mathbb{R}^{N\times N}$.

Therefore, we can solve a linear least-squares problem in the form:

$$\min_{\boldsymbol{u}} \left\| \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{B} \\ \boldsymbol{C} \end{bmatrix} \boldsymbol{u} - \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{w}_1 \\ \boldsymbol{w}_2 \end{bmatrix} \right\|_2^2,$$

where $\boldsymbol{u}, \boldsymbol{f}, \boldsymbol{w}_1$, and $\boldsymbol{w}_2$ are vectorization of $[u_{ij}]$, $[f_{ij}]$, $[w_{1ij}]$, and $[w_{2ij}]$, respectively. *However, the linear least-squares solver (by solving the normal equations, or using the QR decomposition, or using the SVD) has high complexity, leading to significant costs!*

u-subproblem: an FFT-based algorithm

We can use the FFT to solve the u-subproblem:

- Since $K, \partial_1^+, \partial_2^+$ are all discrete convolutions, if we transform the u-subproblem into the Fourier domain, then these operations become element-wise products, e.g., $\mathcal{F}(H \star u) = \mathcal{F}(H) \circ \mathcal{F}(u)$.
- Since the Fourier transform preserves the Frobenius norm, we obtain an equivalent problem (set $\gamma := \beta/\lambda$):

$$\min_u \|\mathcal{F}(H) \circ \mathcal{F}(u) - \mathcal{F}(f)\|_F^2 + \gamma \|\mathcal{F}(\partial_1^+) \circ \mathcal{F}(u) - \mathcal{F}(w_1)\|_F^2 + \gamma \|\mathcal{F}(\partial_2^+) \circ \mathcal{F}(u) - \mathcal{F}(w_2)\|_F^2.$$

- After solving for $\mathcal{F}(u)$ (using first-order optimality condtion), we obtain the solution to the u-subproblem by

$$u = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(H)^* \circ \mathcal{F}(f) + \gamma\left(\mathcal{F}(\partial_1^+)^* \circ \mathcal{F}(w_1) + \mathcal{F}(\partial_2^+)^* \circ \mathcal{F}(w_2)\right)}{\mathcal{F}(H)^* \circ \mathcal{F}(H) + \gamma\left(\mathcal{F}(\partial_1^+)^* \circ \mathcal{F}(\partial_1^+) + \mathcal{F}(\partial_2^+)^* \circ \mathcal{F}(\partial_2^+)\right)} \right),$$

where "$*$" denotes complex conjugacy and the division is element-wise. *Therefore, it requires two ffts and one ifft per iteration.*

Selection of model parameters

- **Noisy level control parameter λ:** An appropriate λ should give a solution u satisfying

$$\|Ku - f\|^2 \approx \|K\bar{u} - f\|^2 = \sigma^2 = Var(n).$$

- **Constraint penalty parameter β:** Parameter β cannot be too small because it would allow $\nabla u = \boldsymbol{w}$ to be violated excessively. However, β cannot be too large either because the larger the β is the less updates applied to $\boldsymbol{w}$ and u, making the algorithm take more iterations. Therefore, we should choose β in a continuation way to balance the speed and accuracy.
- **Prescribed maximum value β_{max}:** The initial value of β is relatively small (e.g., $\beta = 4$). Then β is increased (e.g., doubled) until a prescribed maximum value β_{max} is reached (e,g, 2^{20}).

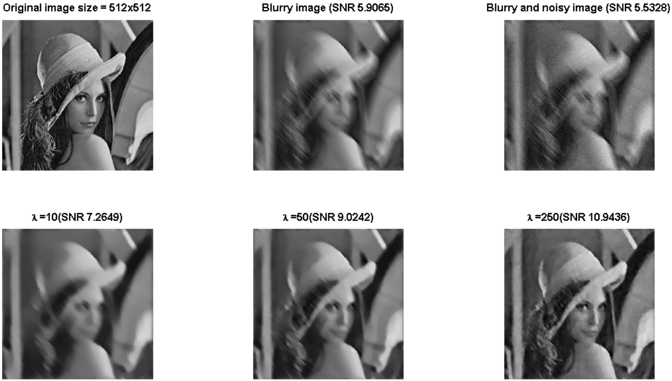
Numerical experiments

% creat a blurring filter

```
>> H = fspecial('motion', 41, 135)
```

% add Gaussian white noise with mean 0 and variance 10^{-3}

```
>> f = imnoise(original, 'gaussian', 0, 1e-3)
```



(All the numerical experiments are performed by Pei-Chiang Shao)

Numerical experiments

% creat a blurring filter

```
>> H = fspecial('gaussian', 41, 10)
```

% add Gaussian white noise with mean 0 and variance 10^{-6}

```
>> f = imnoise(original, 'gaussian', 0, 1e-6)
```

Total variation blind deconvolution

Chan-Wong (1998) formulated the blind deconvolution problem as

$$\min_{u,h} \frac{1}{2}\int_\Omega \big((h\star u)(\boldsymbol{x}) - f(\boldsymbol{x})\big)^2 d\boldsymbol{x} + \alpha_1 \int_\Omega |\nabla u(\boldsymbol{x})| d\boldsymbol{x} + \alpha_2 \int_\Omega |\nabla h(\boldsymbol{x})| d\boldsymbol{x},$$

where the use of TV regularization for the blurring kernel h is due to the fact that some blurring kernels can have edges.

The first-order optimality conditions give

$$u(-\boldsymbol{x}) \star \big((u\star h)(\boldsymbol{x}) - f(\boldsymbol{x})\big) - \alpha_2 \nabla \cdot \left(\frac{\nabla h(\boldsymbol{x})}{|\nabla h(\boldsymbol{x})|}\right) = 0, \quad \boldsymbol{x} \in \Omega,$$
$$h(-\boldsymbol{x}) \star \big((h\star u)(\boldsymbol{x}) - f(\boldsymbol{x})\big) - \alpha_1 \nabla \cdot \left(\frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|}\right) = 0, \quad \boldsymbol{x} \in \Omega,$$

which are the associated Euler-Lagrange equations.

A further study is needed!

References

1. T. F. Chan and C.-K. Wong, Total variation blind deconvolution, *IEEE Transaction on Image Processing,* 7 (1998), pp. 370-375.
2. P. Getreuer, Total variation deconvolution using split Bregman, *Image Processing On Line,* 2 (2012), pp. 158-174.
3. L. I. Rudin, S. Osher, and E. Fatemi, Nonlinear total variation based noise removal algorithms, *Physica D,* 60 (1992), pp. 259-268.
4. Y. Wang, W. Yin, and Y. Zhang, A fast algorithm for image deblurring with total variation regularization, *CAAM Technical Report TR 07-10,* 2007, Rice University.
5. Y. Wang, J. Yang, W. Yin, and Y. Zhang, A new alternating minimization algorithm for total variation image reconstruction, *SIAM Journal on Imaging Sciences,* 1 (2008), pp. 248-272.