Math3360: Mathematical Imaging

Chapter 4: Image Enhancement in the Spatial Domain

In Chapter 4, we have talked about how we can enhance an image in the frequency domain. In this chapter, we will talk about how image enhancement can be done in the spatial domain.

1 Image denoising in the spatial domain

We will first discuss image denoising algorithms in the spatial domain.

1.1 Image denoising by linear filter

Definition 1.1. Linear filter is a process to modify the pixel value by a linear combination of the pixels values of its local neighbourhood.

Example 1.2. Let f be an $N \times N$ image. Extend the image periodically. Modify f to $\tilde{f}$ by:

$$\tilde{f}(x,y) = f(x,y) + 3f(x-1,y) + 2f(x+1,y).$$

This is a linear filter.

Example 1.3. Define

$$\tilde{f}(x,y) = \frac{1}{4}\left(f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)\right)$$

This is also a linear filter.

Linear filter versus discrete convolution

We consider an image f to be defined on $[-M,M] \times [-N,N]$. So, f is a $(2M+1) \times (2N+1)$ image.

Assume f is also periodically extended. Let H be another $(2M+1) \times (2N+1)$ matrix defined on $[-M,M] \times [-N,N]$.

Recall: The discrete convolution is defined as:

$$I * H(u,v) = \sum_{m=-M}^{M} \sum_{n=-N}^{N} I(u-m, v-n)H(m,n)$$

(Linear combination of pixel values around (u,v))

Therefore, **Linear filter is equivalent to a discrete convolution**.

Remark. *1. $DFT(I * H) = C \cdot DFT(I) \cdot DFT(H)$ for some constant C. Therefore, the linear filter is equivalent to modifying the DFT coefficients by multiplication.*

2. *A geometric illustration of the idea is as follows.*

$$\begin{array}{ccc} I & & \hat{I} \\ \downarrow * H & \cong & \downarrow \times \hat{H} \\ \tilde{I} & \underset{iDFT}{\overset{DFT}{\rightleftarrows}} & \hat{\tilde{I}} \\ \textit{(spatial)} & & \textit{(frequency)} \end{array}$$

Example 1.4. In Example 1.2, if f is defined on $[-M, M] \times [-N, N]$, then:

$$\tilde{f} = f * H$$

where

$$H = \begin{pmatrix} 0 & 2 & 0 \\ 0 & 1 & 0 \\ 0 & 3 & 0 \end{pmatrix}$$

In Example 1.3, $\tilde{f} = f * H$ where

$$H = \frac{1}{4} \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Remark. *H is called the filter.*

Some commonly used linear filters:

- **Mean filter:**

$$H = \frac{1}{9} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

(Here, we only write down the entries of the matrix for indices $-1 \leq k, l \leq 1$ for simplicity. All other matrix entries are equal to 0.)

This is called the *mean filtering with window size* 3×3.

Below is an example of mean filtering on an image with impulse noise:

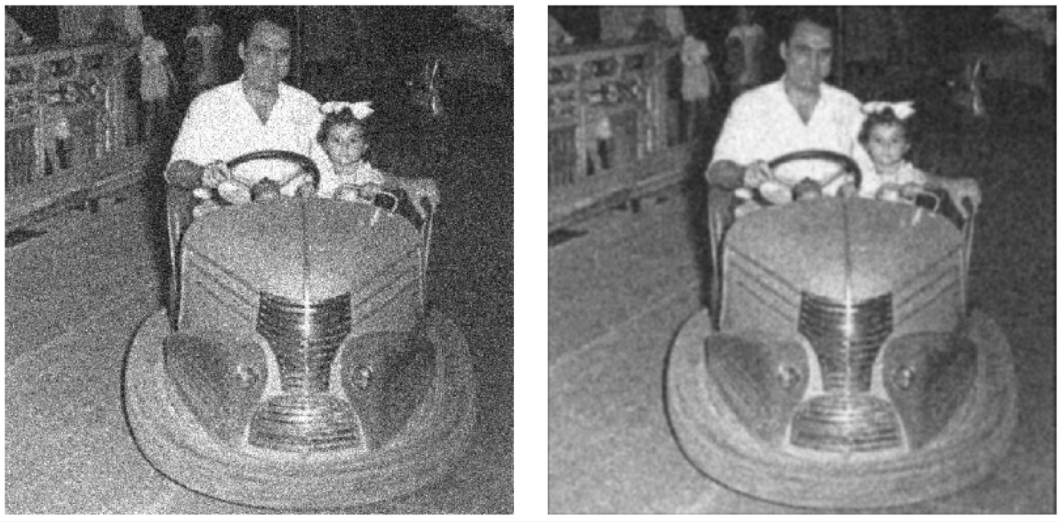
Impulse noise

After mean filter

Below is an example of mean filtering on an image with Gaussian noise:

Gaussian noise | After mean filter

- **Gaussian filter:** The entries of H are given by the Gaussian function $g(r) = exp\left(-\frac{r^2}{2\sigma^2}\right)$, where $r = \sqrt{x^2 + y^2}$.
  Below is an example of Gaussian filtering on an image:



Real image | After Gaussian filter

(Please refer to "Lecture 17 powerpoint" for some more examples of mean filter and Gaussian filter on real images)

Properties of linear filtering

- **Associativity:** $A * (B * C) = (A * B) * C$
- **Commutativity:** $I * H = H * I$
- **Linearity:**

$$(s \cdot I) * H = I * (s \cdot H) = s \cdot (I * H)$$
$$(I_1 + I_2) * H = (I_1 * H) + (I_2 * H)$$

Remark. *1. Advantage of Gaussian filter: We can check that convolution of Gaussian filter is again Gaussian (with larger σ).*

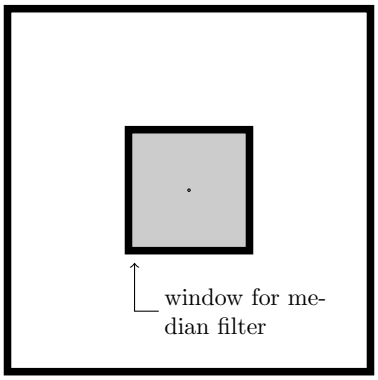
2. Thus, successive Gaussian filter = Gaussian filter with a larger σ (because of the property of associativity).

Proof of associativity

$$\begin{aligned}
[(x*y)*z](n) &= \sum_{m=0}^{N-1} \widehat{(x*y)*z}(m) e^{j2\pi\frac{mn}{N}} \\
&= \sum_{m=0}^{N-1} C\widehat{x*y}(m)\hat{z}(m)e^{j2\pi\frac{mn}{N}} \\
&= \sum_{m=0}^{N-1} C^2\hat{x}(m)\hat{y}(m)\hat{z}(m)e^{j2\pi\frac{mn}{N}} \\
&= \sum_{m=0}^{N-1} C\hat{x}(m)\widehat{y*z}(m)e^{j2\pi\frac{mn}{N}} \\
&= \sum_{m=0}^{N-1} \widehat{x*(y*z)}(m)e^{j2\pi\frac{mn}{N}} \\
&= (x*(y*z))(n)
\end{aligned}$$

1.2 Generalization of linear filter: Non-linear (spatial) filter

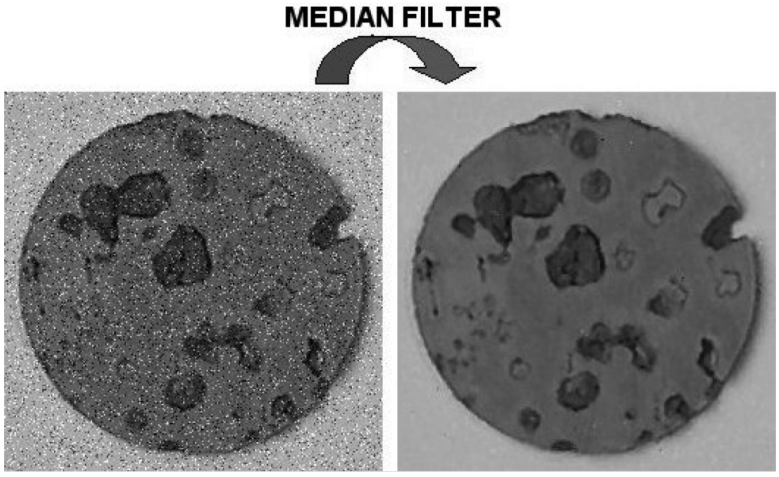
Median filter



Take a window with center at pixel (x_0, y_0). Update the pixel value at (x_0, y_0) from $I(x_0, y_0)$ to $\tilde{I}(x_0, y_0) = \text{median}(I \text{ within the window})$

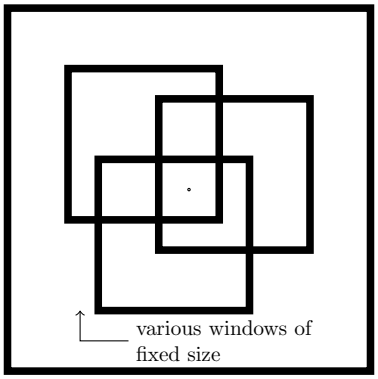
Example 1.5. If the pixel values within a window are $0, 0, 1, 2, 3, 7, 8, 9, 9$, then the pixel value is updated as 3 (median).

Below is an example of median filtering on an image:



Below is a comparison of mean and median filtering on an image:

Edge-preserving filter



- **Step 1:** Consider all windows with certain size around pixel (x_0, y_0) (not necessarily centered

at (x_0, y_0));

- **Step 2:** Select a window with minimal variance;
- **Step 3:** Do a linear filter (mean filter, Gaussian filter and so on).

Below is an example of edge preserving filtering on an image (compared with other types of filtering):

(a) Original "Greek Flag"

(b) Flat filter

(c) Edge preserving flat

(d) Gaussian filter

(e)Edge preserving Gaussian

Non-local mean filter

Let g be a $N \times N$ image. Let $X = (x, y)$ and $X' = (x', y')$ be two different pixels of g. Consider two corresponding neighbourhoods:

$$S_X = \{(x+s, y+t) : -a \leq s, t \leq a\}\,;\ S_{X'} = \{(x'+s, y'+t) : -a \leq s, t \leq a\}$$

$\therefore$ Sizes of S_X and $S_{X'}$ are both $\underbrace{(2a+1)}_{=m} \times \underbrace{(2a+1)}_{=m}$.

Denote $g_X = g|_{S_X}$ and $g_{X'} = g|_{S_{X'}}$, then g_X and $g_{X'}$ are two $m \times m$ small images.
We call g_X and $g_{X'}$ the local patch of g at X and X' respectively.

Apply Gaussian filter (linear) to g_X and $g_{X'}$ to get $\tilde{g}_X$ and $\tilde{g}_{X'}$

Define the least square distance between two local patches as:

$$||\tilde{g}_X - \tilde{g}_{X'}||^2 = \text{ sum of squares of coefficients of the matrix } (\tilde{g}_X - \tilde{g}_{X'})$$

Define the weight by : $w(X, X') = \exp\left(\dfrac{-||\tilde{g}_X - \tilde{g}_{X'}||^2}{h^2}\right)$, where h is called the noise level parameter.

Definition 1.6. The **non-local mean filter** of an image g is given by:

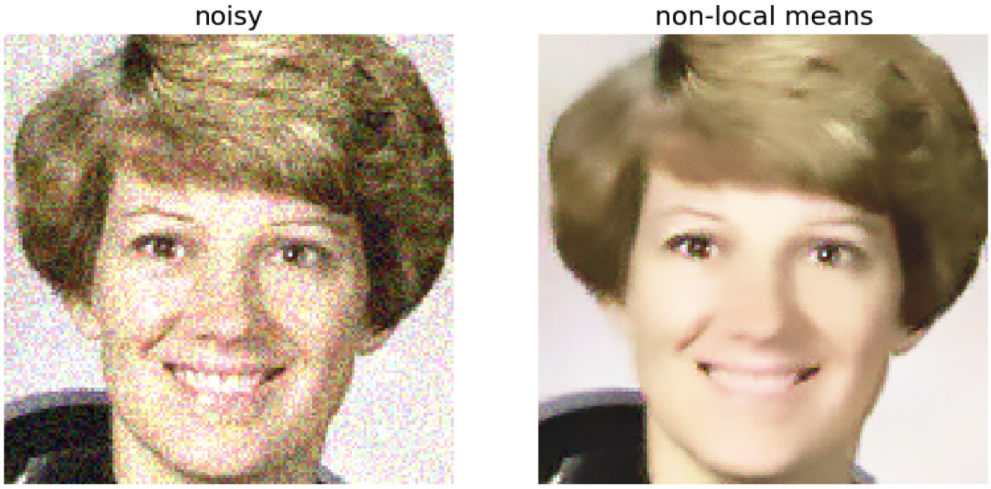
$$\hat{g} = \frac{\sum_{X' \in \text{ image domain}} w(X, X') g(X')}{\sum_{X' \in \text{ image domain}} w(X, X')}$$

where $\hat{g}$ is the output image.

Remark:

- The image g is often: 1. Periodically extended; or 2. set outside region to have zero pixel values.
- The weight is smaller if the overall intensities over a local patch at X and X' are different.

Below is an example of non-local mean filtering on an image:



(Please see "Lecture 18 powerpoint" for more illustration of non-local mean filter on real images)

1.3 Image denoising by Anisotropic Diffusion

Consider the following partial differential equation:

$$\begin{aligned}\frac{\partial I(x,y;\sigma)}{\partial \sigma} &= \sigma\left[\frac{\partial^2 I(x,y;\sigma)}{\partial x^2}+\frac{\partial^2 I(x,y;\sigma)}{\partial y^2}\right]\\ &= \sigma\nabla\cdot(\nabla I(x,y;\sigma 0)) \qquad (*)\end{aligned}$$

(where $\nabla\cdot$ = divergence defined by $\nabla\cdot((v_1,v_2)) = \dfrac{\partial v_1}{\partial x}+\dfrac{\partial v_2}{\partial y}$ and

∇I = gradient of $I = \left(\dfrac{\partial I}{\partial x},\dfrac{\partial I}{\partial y}\right)$)

Then, the Gaussian function with standard deviation σ,

$$g(x,y;\sigma) = \frac{1}{2\pi\sigma^2}exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

satisfies equation (*).

In fact, the Gaussian linear filter is approximately solving the heat diffusion equation.

Given an image $I(x,y)$ (which is assumed to be continuous image over the whole 2D domain), the Gaussian filter is equivalent to the convolution of I with the Gaussian function:

$$\begin{aligned}\widetilde{I}(x,y;\sigma) &= I * g(x,y;\sigma)\\ &:= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x-u,y-v;\sigma)I(u,v)\ du\ dv\\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(u,v;\sigma)I(x-u,y-v)\ du\ dv\end{aligned}$$

This is analogous to the discrete convolution:

$$I * J(u,v) = \sum_m \sum_n I(u-m, v-n)J(m,n)$$

$$\begin{aligned}
\therefore \frac{\partial \widetilde{I}}{\partial \sigma} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{\partial g(u,v;\sigma)}{\partial \sigma} I(x-u, y-v)\ du\ dv \\
&= \sigma \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{\partial^2 g(u,v;\sigma)}{\partial u^2} I(x-u, y-v)\ du\ dv \\
&+ \sigma \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{\partial^2 g(u,v;\sigma)}{\partial v^2} I(x-u, y-v)\ du\ dv \\
&= \sigma \frac{\partial^2}{\partial x^2} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(u,v;\sigma) I(x-u, y-v)\ du\ dv \\
&+ \sigma \frac{\partial^2}{\partial y^2} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(u,v;\sigma) I(x-u, y-v)\ du\ dv \\
&= \sigma \nabla \cdot (\nabla \widetilde{I}(x,y;\sigma))
\end{aligned}$$

Remark. • *Here, we us the fact that:*

$$\frac{\partial}{\partial x}(g(x,y;\sigma) * f) = \frac{\partial g(x,y;\sigma)}{\partial x} * f$$

- *Gaussian filtering ≈ solving the above partial differential equation.*
- *Refer to powerpoint for the solution of the heat diffusion equation at different scales*

Anisotropic diffusion for edge-preserving image denoising

The general form of a heat diffusion equation can be written as:

$$\frac{\partial I(x,y;\sigma)}{\partial \sigma} = \nabla \cdot (K(x,y)\nabla I(x,y;\sigma))$$

where K controls the rate of diffusion at (x,y) (smaller K = smaller diffusion at (x,y)).

Motivation: Less diffusion on the edges.

Edge detector: Edges of an image can be determined by $|\nabla I(x,y)|$.

If (x,y) is on the edges, $|\nabla I(x,y)|$ is big while if (x,y) is in the interior of the object, $|\nabla I(x,y)| \approx 0$ (Assume the image is piecewise constant)

Suitable K to preserve edges:

1. $K(x,y) = \dfrac{1}{|\nabla I(x,y)|}$ (not good as ∇I can be $\vec{0}$)
   Modification: $K(x,y) = \dfrac{1}{|\nabla I(x,y)| + \varepsilon^2}$ for ε small
2. $K(x,y) = exp\left(-\dfrac{|\nabla I(x,y;\sigma)|}{b}\right)$

Remark. *We can choose K such that K depends on $|\nabla I|$ and $K \approx 0$ when $|\nabla I|$ is big.*

Hence, the anisotropic diffusion algorithm for solving the image de-noising problem can be written as:

$$\frac{\partial I(x,y;\sigma)}{\partial \sigma} = \nabla \cdot \left(exp\left(-\frac{|\nabla I(x,y;\sigma)|}{b}\right)\nabla I(x,y;\sigma)\right) \qquad (**)$$

In the discrete case, (**) can be considered as:

$$I^{n+1}(x,y) - I^n(x,y) = \mathcal{D}_1\left(exp\left(-\frac{|\nabla I^n(x,y)|}{b}\right)\mathcal{D}_2 I^n(x,y)\right)$$

where $\mathcal{D}_1$ *= linear operator approximating* $\nabla \cdot$ *and*
$\mathcal{D}_2$ *= linear operator approximating the gradient* ∇

Starting with $I^0(x, y) = I(x, y)$, which is the original image, we iteratively de-noise the original image $I(x, y) := I^0(x, y)$. Such a process is called **anisotropic diffusion image denoising**.

Below is an example of (edge-preserving) anisotropic diffusion on an image:

Seven iterations

Fourteen iterations

Twenty iterations

1.4 Image denoising via energy minimization

Let g be the noisy image. We will consider additive noise. So, g can be written as:

$$g(x, y) = f(x, y) + n(x, y)$$

where $f(x, y)$ is the clean image and $n(x, y)$ is the noise.

Recall: Laplacian marking: $g = f - \Delta f$ obtains a sharp (non-smooth) image from a smooth image.

Conversely, to get a smooth image f from a non-smooth image g, we can solve f such that: $f = g + \Delta f$ (Partial differential equation)

The equation can be discretized to get:

$$f(x, y) = g(x, y) + [f(x + 1, y) + f(x - 1, y) + f(x, y + 1) + f(x, y - 1) - 4f(x, y)] \qquad (***)$$

for all (x, y). (which is a linear system)

The above linear system can be solved by direct method (with big matrix) or by iterative method.

Simple iterative scheme: Let g be an $N \times N$ image.
Step 1: Let $f^0(x, y) = g(x, y)$ (Initial guess of the solution)

Step 2: For $n \geq 0$ and for all $(x, y), x = 1, \cdots, M, y = 1, \cdots, N$,

$$f^{n+1}(x,y) = g(x,y) + [f^n(x+1,y) + f^n(x-1,y) + f^n(x,y+1) + f^n(x,y-1) - 4f^n(x,y)]$$

Impose boundary conditions by reflection:

$$\begin{aligned}
&f^{n+1}(0,y) = f^{n+1}(2,y); \quad f^{n+1}(M+1,y) = f^{n+1}(M-1,y) \quad \text{for } y = 1, \cdots, N \\
&f^{n+1}(x,0) = f^{n+1}(x,2); \quad f^{n+1}(x,N+1) = f^{n+1}(x,N-1) \quad \text{for } x = 1, \cdots, M \\
&f^{n+1}(0,0) = f^{n+1}(2,2); \quad f^{n+1}(0,N+1) = f^{n+1}(2,N-1) \\
&f^{n+1}(M+1,0) = f^{n+1}(M-1,2); \quad f^{n+1}(M+1,N+1) = f^{n+1}(M-1,N-1)
\end{aligned}$$

(similar for f^n)

Step 3: Continue the process until $||f^{n+1} - f^n|| \leq$ tolerance. (Convergence depends on the spectral radius of a matrix)

In fact, the solution of (***) is a minimizer of an energy (taking into account the extension by reflection outside the domain):

$$E_{discrete}(f) = \sum_{x=1}^{M}\sum_{y=1}^{N}(f(x,y) - g(x,y))^2 + \sum_{x=1}^{M}\sum_{y=1}^{N}[(f(x+1,y) - f(x,y))^2 + (f(x,y+1) - f(x,y))^2]$$

Suppose f is a minimizer of $E_{discrete}$. Then, for each position (x, y),

$$\begin{aligned}
0 &= \frac{\partial E_{discrete}}{\partial f(x,y)} \\
&= 2(f(x,y) - g(x,y)) + 2(f(x+1,y) - f(x,y))(-1) + 2(f(x,y+1) - f(x,y))(-1) \\
&+ 2(f(x,y) - f(x-1,y)) + 2(f(x,y) - f(x,y-1))
\end{aligned}$$

After rearrangement,

$$f(x,y) = g(x,y) + [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)]$$

The continuous version of $E(f)$ is:

$$E(f) = \iint (f(x,y) - g(x,y))^2 \, dx \, dy + \iint \left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2 \, dx \, dy$$

or

$$E(f) = \iint (f(x,y) - g(x,y))^2 \, dx \, dy + \iint |\nabla f|^2 \, dx \, dy$$

Remark:

- Solving $-\Delta f + f = g$ is equivalent to solving an energy minimization problem.
- First term aims to find a smooth image, which is close enough to g (original image).
- Second term is called the *regularization term*, which is to minimize the derivative of f (Enhance smoothness).
- $-\nabla f + f = g$ can be solved in the frequency domain

$$\begin{aligned}
&DFT(f) = DFT(g + \Delta f) = DFT(g + l * f) \\
\Leftrightarrow\ &DFT(f)(u,v) = DFT(g)(u,v) + cDFT(l)(u,v)DFT(f)(u,v) \\
\Leftrightarrow\ &DFT(f)(u,v) = \frac{1}{1 - cDFT(l)(u,v)} DFT(g)(u,v)
\end{aligned}$$

  The inverse Fourier transform can then be applied.

1.5 Image denoising by solving PDE

Consider the harmonic-L2 image denoising model:

$$E(f) = \int_\Omega (f(x,y) - g(x,y))^2 \, dx \, dy + \int |\nabla f|^2 \, dx \, dy$$

We find f that minimizes $E(f)$. Take any function $v(x,y)$, and consider

$$\begin{aligned} s(\varepsilon) &= E(f + \varepsilon v) \\ &= \int_\Omega (f(x,y) + \varepsilon v(x,y) - g(x,y))^2 \, dx \, dy + \int_\Omega |\nabla f + \varepsilon \nabla v|^2 \, dx \, dy \\ \frac{d}{d\varepsilon} s(\varepsilon) &= 2 \int_\Omega (f(x,y) + \varepsilon v(x,y) - g(x,y)) v(x,y) \, dx \, dy \\ &\quad + 2 \int_\Omega \left[\left(\frac{\partial f}{\partial x} + \varepsilon \frac{\partial v}{\partial x} \right) \frac{\partial v}{\partial x} + \left(\frac{\partial f}{\partial y} + \varepsilon \frac{\partial v}{\partial y} \right) \frac{\partial v}{\partial y} \right] \, dx \, dy \end{aligned}$$

If f is the minimizer, then $\left. \frac{d}{d\varepsilon} \right|_{\varepsilon=0} s(\varepsilon) = 0$

$$\therefore s'(0) = 0 = 2 \int_\Omega (f(x,y) - g(x,y)) v(x,y) \, dx \, dy + 2 \int_\Omega (f_x v_x + f_y v_y) \, dx \, dy$$

Recall: (Integration by parts)

$$\int_\Omega \nabla f \cdot \nabla g \, dx \, dy = - \int_\Omega (\nabla \cdot \nabla f) g \, dx \, dy + \int_{\partial\Omega} g (\nabla f \cdot \vec{n}) \, ds$$

where $\vec{n} = (n_1, n_2)$ = outward normal on the boundary.

$$\therefore 0 = \int_\Omega (f - g) v \, dx \, dy - \int_\Omega (\nabla \cdot \nabla f) v \, dx \, dy - \int_{\partial\Omega} (\nabla f \cdot \vec{n}) v \, ds$$

Overall, we get

$$\int_\Omega (f - g - \Delta f) v \, dx \, dy - \int_{\partial\Omega} (\nabla f \cdot \vec{n}) v \, ds = 0$$

from where we obtain:

$$\begin{cases} f - g - \Delta f = 0 & \text{in } \Omega \\ \nabla f \cdot \vec{n} = 0 & \text{on } \partial\Omega \end{cases} \qquad \text{(PDE)}$$

Conversely, given f such that the above PDE is satisfied, for any other h

$$\begin{aligned} &E(h) - E(f) = \int_\Omega [(h-g)^2 - (f-g)^2 + |\nabla h|^2 - |\nabla f|^2] \, dx \, dy \\ &= \int_\Omega [(h-g) - (f-g)]^2 + |\nabla h - \nabla f|^2 + 2 \nabla f \cdot (\nabla h - \nabla f) + 2(f-g)(h-f)] \, dx \, dy \\ &\geq \int_\Omega 2 \nabla f \cdot \nabla (h-f) + 2(f-g)(h-f) \, dx \, dy \\ &= 2 \int_\Omega - \underbrace{(\nabla \cdot \nabla f)(h-f) + (f-g)(h-f)}_{=0} \, dx \, dy + 2 \int_{\partial\Omega} (\cancelto{0}{\nabla f \cdot \vec{n}})(h-f) = 0 \end{aligned}$$

Therefore, $E(h) \geq E(f)$ and f is the minimizer.

Remark.

- *Energy minimization is usually done by:*
  1. *Iterative method / Gradient descent*
  2. *Solving PDE*

- *Anisotropic diffusion is related to minimizing:*

$$E(f) = \int_\Omega K(x,y)|\nabla f(x,y)|^2 \, dx \, dy$$

Intuitively, we want to minimize "derivative" / "jump" $|\nabla f(x,y)|$. On edges, $K(x,y)$ is small, so contribution of $|\nabla f(x,y)|$ on the edges is small and thus less minimization of $|\nabla f(x,y)|$ on the edges.

- *Energy minimization approaches are commonly called variational image denoising.*

1.6 Total variation (TV) denoising

Other name: ROF model: Rudin, Osher, Fatemi

Motivation: The previous model solves $f = g + \Delta f$, given a noisy image g. Δf may smooth out edges. Therefore, we consider $\nabla \cdot (K(x,y)f(x,y))$ where $K(x,y)$ is small if (x,y) is on the boundary/edges.

Goal: Given a noisy image $g(x,y)$, we look for $f(x,y)$ that solves:

$$f = g + \frac{\partial}{\partial x}\left(\frac{1}{|\nabla f|(x,y)}\frac{\partial f}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{1}{|\nabla f|(x,y)}\frac{\partial f}{\partial y}\right) \qquad (***)$$

Remark. *Again, a problem arises when $|\nabla f|(x,y) = 0$. We will take care of it later.*

We will show that the PDE (****) must be satisfied by a minimizer of the following energy functional:

$$J(f) = \frac{1}{2}\iint_\Omega (f(x,y) - g(x,y))^2 \, dx \, dy + \lambda \iint_\Omega |\nabla f|(x,y) \, dx \, dy$$

where Ω is the image domain.

To do this, we must have "$\frac{\partial J}{\partial f} = 0$". Let's discretize $J(f)$. The discrete version of $J(f)$ is

$$J(f) = \frac{1}{2}\sum_{x=1}^{N}\sum_{y=1}^{N}(f(x,y) - g(x,y))^2 + \lambda \sum_{x=1}^{N}\sum_{y=1}^{N}\sqrt{(f(x+1,y) - f(x,y))^2 + (f(x,y+1) - f(x,y))^2}$$

(Finite difference approximation of $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$ with $\Delta x = \Delta y = 1$)

Therefore,

$$\begin{aligned}\frac{\partial J}{\partial f(x,y)} = (f(x,y) - g(x,y)) &+ \lambda \frac{2(f(x+1,y) - f(x,y))(-1) + 2(f(x,y+1) - f(x,y))(-1)}{2\sqrt{(f(x+1,y) - f(x,y))^2 + (f(x,y+1) - f(x,y))^2}} \\ &+ \lambda \frac{2(f(x,y) - f(x-1,y))}{2\sqrt{(f(x,y) - f(x-1,y))^2 + (f(x-1,y+1) - f(x-1,y))^2}} \\ &+ \lambda \frac{2(f(x,y) - f(x,y-1))}{2\sqrt{(f(x+1,y-1) - f(x,y-1))^2 + (f(x,y) - f(x,y-1))^2}} = 0\end{aligned}$$

By simplification, we get:

$$\begin{aligned}f(x,y) - g(x,y) = \lambda &\left\{ \frac{f(x+1,y) - f(x,y)}{\sqrt{(f(x+1,y) - f(x,y))^2 + (f(x,y+1) - f(x,y))^2}} \right. \\ &\left. - \frac{f(x,y) - f(x-1,y)}{\sqrt{(f(x,y) - f(x-1,y))^2 + (f(x-1,y+1) - f(x-1,y))^2}} \right\} \\ + \lambda &\left\{ \frac{f(x,y+1) - f(x,y)}{\sqrt{(f(x+1,y) - f(x,y))^2 + (f(x,y+1) - f(x,y))^2}} \right. \\ &\left. - \frac{f(x,y) - f(x,y-1)}{\sqrt{(f(x+1,y-1) - f(x,y-1))^2 + (f(x,y) - f(x,y-1))^2}} \right\}\end{aligned}$$

for all $1 \leq x, y \leq N$.

(Discretization of $f - g = \lambda \nabla \cdot \left(\frac{\nabla f}{|\nabla f|} \right)$.)

The above equation is a non-linear equation. (Difficult to solve!!) We apply the gradient descent method to solve it iteratively.

General gradient descent method

We consider the general case to solve:

$$\min_f J(f)$$

where f can be a number, a vector or a function. In our case, f is a discrete image function $(x, y) \mapsto f(x, y)$. Hence, we consider J to be a function defined on a discrete image, which depends on $M \times N$ variables (assuming the image is of size $M \times N$).

Consider a time-dependent image $f(x, y; t)$. Assume $f(x, y; t)$ solve the ODE:

$$\frac{df(x, y; t)}{dt} = -\nabla J(f(x, y; t)) \text{ (gradient descent equation)}$$

We show that $J(f(x, y; t))$ is decreasing as t increases.

In fact,

$$\frac{d}{dt} J(f(x, y; t)) = \nabla J(f(x, y; t)) f'(x, y; t) = -|\nabla J(f(x, y; t))|^2 \leq 0$$

Therefore, $J(f(x, y; t))$ is decreasing as t increases. We solve (*) and $f(x, y; t) \to \bar{f}(x, y)$ is the minimizer.

We solve the gradient descent equation iteratively. Let $f^0(x, y)$ be the initial guess. (Here $1 \leq x \leq M,\ 1 \leq y \leq N$ and extend f^0 by reflection.)

We solve (*) in a discrete sense:

$$\frac{f^{n+1} - f^n}{\Delta t} = -\nabla J(f^n) \qquad (*****)$$

Δt is called the time step, which must be chosen carefully.

In our case,. (*****) becomes:

$$\begin{aligned}
&\frac{f^{n+1}(x, y) - f^n(x, y)}{\Delta t} \\
&= -(f^n(x, y) - g(x, y)) + \lambda \frac{f^n(x+1, y) - f^n(x, y)}{\sqrt{(f^n(x+1, y) - f^n(x, y))^2 + (f^n(x, y+1) - f^n(x, y))^2}} \\
&- \lambda \frac{f^n(x, y) - f^n(x-1, y)}{\sqrt{(f^n(x, y) - f^n(x-1, y))^2 + (f^n(x-1, y+1) - f^n(x-1, y))^2}} \\
&+ \lambda \frac{f^n(x, y+1) - f^n(x, y)}{\sqrt{(f^n(x+1, y) - f^n(x, y))^2 + (f^n(x, y+1) - f^n(x, y))^2}} \\
&- \lambda \frac{f^n(x, y) - f^n(x, y-1)}{\sqrt{(f^n(x+1, y-1) - f^n(x, y-1))^2 + (f^n(x, y) - f^n(x, y-1))^2}}
\end{aligned}$$

subject to the boundary condition imposed by reflection.

In the continuous setting, it is equivalent to:

$$\frac{df}{dt} = -(f - g) + \lambda \nabla \cdot \left(\frac{\nabla f}{|\nabla f|} \right)$$

Below is an example of TV/ROF denoising on an image:

2 Image deblurring in the spatial domain (Optional)

(Spatial) De-blurring de-noising variational model

Let $g = h * f + n$ be a blurred image, where h is the degradation functional and n is the noise. Assume f is the original clean image, which is continuous on $\mathbb{R}^2$. We consider the variational model to find f that minimizes:

$$J(f) = \frac{1}{2}\iint_{\mathbb{R}^2} (h * f(x,y) - g(x,y))^2 \, dx \, dy \; + \lambda \iint_{\mathbb{R}^2} |\nabla f| \, dx \, dy$$

Remark. *It is the TV de-noising model with image blur incorporated. We proceed to solve the optimization problem by steepest (gradient) descent method.*

Observation: Let $Hf(x,y) := h * f(x,y) = \displaystyle\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\alpha,\beta) f(x-\alpha, y-\beta) \, dx \, dy.$
Let $\langle f, g\rangle = \displaystyle\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)g(x,y) \, dx \, dy.$ Then:

$$\langle Hf, g\rangle = \langle f, H^* g\rangle$$

where H^* is the adjoint of H. We want to compute H^*.

$$\begin{aligned}
\langle Hf, g\rangle &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h * f(x,y) g(x,y) \, dx \, dy \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\alpha,\beta) f(x-\alpha, y-\beta) \, d\alpha \, d\beta\right] g(x,y) \, dx \, dy \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x-\alpha, y-\beta) g(x,y) \, dx \, dy \right] h(\alpha,\beta) \, d\alpha \, d\beta
\end{aligned}$$

Let $X = x - \alpha, Y = y - \beta$, we obtain

$$\begin{aligned}
\langle Hf, g\rangle &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(X,Y) g(X+\alpha, Y+\beta) \, dX \, dY \right] h(\alpha,\beta) \, d\alpha \, d\beta \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(X,Y) \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\alpha,\beta) g(\alpha+X, \beta+Y) \, d\alpha \, d\beta\right] \, dX \, dY
\end{aligned}$$

Let $a = -\alpha, b = -\beta$, we obtain

$$\begin{aligned}
\langle Hf, g\rangle &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(X,Y) \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(-a,-b) g(X-a, Y-b)(-1)(-1) \, da \, db \right] \, dX \, dY \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(X,Y) \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \tilde{h}(a,b) g(X-a, Y-b) \, da \, db \right] \, dX \, dY
\end{aligned}$$

where $\tilde{h}(a,b) = h(-a,-b)$. Hence,

$$\langle Hf, g\rangle = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(X,Y)\tilde{h} * g(X,Y)\ dX\ dY\ = \langle f, H^*g\rangle.$$

Therefore, $\boxed{H^*g(x,y) = \tilde{h} * g(x,y)}$.

Now, gradient descent algorithm can be done as follows.

Let $s(\varepsilon) = J(f + \varepsilon w)$, then $\left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0} s(\varepsilon) = 0$. Then:

$$\begin{aligned} s'(0) &= \frac{1}{2}\iint \left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0} (h*f + \varepsilon h*w - g)^2\ dx\ dy\ + \lambda \iint \left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0} |\nabla f + \varepsilon\nabla w|\ dx\ dy \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (h*f - g)h*w\ dx\ dy\ - \lambda \iint \nabla\cdot\left(\frac{1}{|\nabla f|}\nabla f\right) w\ dx\ dy \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left[\tilde{h} * (h*f - g) - \lambda\nabla\cdot\left(\frac{1}{|\nabla f|}\nabla f\right)\right] w\ dx\ dy \end{aligned}$$

In order to minimize J, the descent direction should be:

$$w = -\tilde{h} * (h*f - g) + \lambda\nabla\cdot\left(\frac{1}{|\nabla f|}\nabla f\right)$$

Hence, the gradient descent algorithm is given by:

$$\frac{f^{n+1} - f^n}{dt} = -\tilde{h} * (h*f^n - g) + \lambda\nabla\cdot\left(\frac{1}{|\nabla f^n|}\nabla f^n\right)$$

with dt the time step and suitable parameter added to $\nabla\cdot\left(\frac{1}{|\nabla f^n|}\nabla f^n\right)$ to avoid singularity.

Remark. *1. Here, we consider the continuous image to be defined on the whole domain $\mathbb{R}^2$.*

2. In practice, if we are given an image defined on a compact domain Ω, we extend the image to the whole $\mathbb{R}^2$ by setting the region outside Ω to be zero.

3. Let TV-deblurring-denoising model simultaneously deblur an image and also denoise the image, which preserves edges.

Further remark: In the discrete case, we may consider finding $f(x,y)$ $(0 \le x, y \le N-1)$ which minimizes:

$$E(f(x,y)) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} (h*f(x,y) - g(x,y))^2 + \alpha \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} |\nabla f(x,y)|$$

We can again derive the gradient descent iterative scheme to minimize $E(f)$.

Again, we can observe the following.

Let $Hf = h*f$. Define: $\langle f, g\rangle = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)g(x,y)$.

Then, $\langle H(f), g\rangle = \langle f, H^*g\rangle$ where H^* is the adjoint of H.

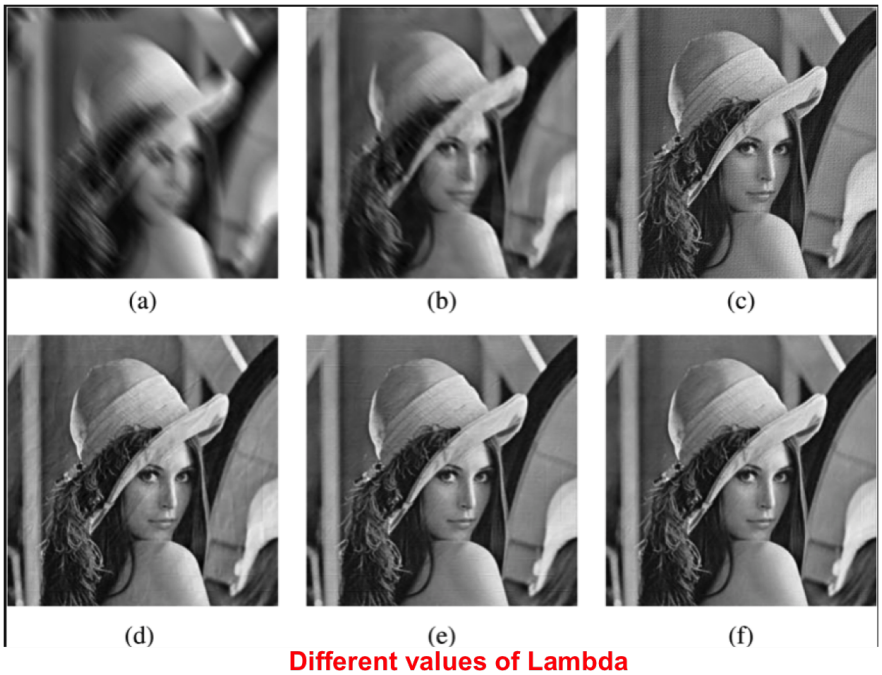
Similarly, we can find H^T as follows.

$$\begin{aligned} \langle Hf, g\rangle &= \sum_{x=0}^{N-1}\sum_{y=0}^{N-1}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} h(x-m, y-n)f(m,n)g(x,y) \\ &= \sum_{m=0}^{N-1}\sum_{n=0}^{N-1}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} h(x-m, y-n)g(x,y)f(m,n) \\ &= \sum_{m=0}^{N-1}\sum_{n=0}^{N-1}\left(\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}\tilde{h}(m-x, n-y)g(x,y)\right) f(m,n) \end{aligned}$$

where $\tilde{h}(a,b) = h(-a,-b)$.

$$\therefore \langle Hf, g\rangle = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)\tilde{h} * g(m,n)$$

Below is an example of TV/ROF deblurring on an image (with different values of λ):



Different values of Lambda

3 Image sharpening in the spatial domain (Optional)

The image sharpening in the spatial domain is exactly the same as that in the frequency domain. Instead of working on the frequency domain, we work on the spatial domain directly.

Let f be an input image. To sharpen the image, we compute a smoother image (by Gaussian filtering or mean filtering) f_{smooth}. Define the sharper image g as:

$$g(x,y) = f(x,y) + k(f(x,y) - f_{smooth}(x,y))$$

When $k = 1$, the method is called unsharp masking.
When $k > 1$, the method is called highboost filtering.