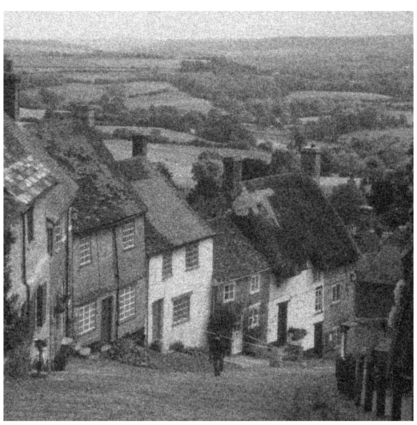
Variational Method for Image Denoising

Joel Kim, Grace Pan, Kai Wang, Mark Xu (Group 10)
{joel.kim—, g4pan, k282wang, chteenghong.xu}@uwaterloo.ca
University of Waterloo
Waterloo, ON, Canada

January 10, 2021

Our group is interested in applying Calculus of Variation in Image Processing. In mathematics, an image is a function that maps every point in some domain to of definition to a particular color value. Generally speaking, mathematical image processing deals with the following tasks: denoising, image decomposition, deblurring, etc. Many image processing problems can be reduced to finding the minimizing function, which is where the variational method can be applied (e.g., A representative example would be using non-local variational methods in image restoration). We found different textbooks and papers ((Aubert and Kornprobst, 2006)) ((Bredies and Lorenz, 2018)) ((Chambolle, 2004)) ((Chambolle et al., 2010)) ((Hecht, 2002)) ((Rudin et al., 1992)) that provide a detailed introduction to image processing and algorithm online ((dog, 2014)) for the implementation part. Our project focuses on the denoising task and how the Calculus of Variation can help solve it.



Noisy image

After denoising

Contents

1 Background

1.1 Brief History of Images

Images are ubiquitous in the modern world. Over time, the methods to produce and store these images has evolved. The first images or photographs were first created by projecting a real-world scene onto a two-dimensional image plane. After extended exposure of the background to an image plane, they were processed with chemicals to map the different brightness and color from the scene to the image plane. These early photographs were monochromatic, and hence the color was mapped to a grayscale. Later on, colored photographs were developed, which could map the scenes' colors to an image, just like how humans can see colors.

Today, there are numerous methods of imaging using electromagnetic radiation and digital technologies. X-ray imaging takes advantage of dense objects like bones to absorb x-rays while soft tissues like skin do not. This allows an image to develop, which shows dense objects that absorb x-rays while not showing objects where x-ray passes through. Imaging methods like computer tomography (CT) use x-rays to scan objects in different directions. Then, this data is used to reconstruct a three-dimensional image of the density of the object.

1.2 Mathematical Definition of Images

Mathematically, an image is "a function that maps every point in some domain of definition to a certain color value." ((Aubert and Kornprobst, 2006))

In other words, an image u is a mapping from an $image\ domain$ Ω to some $color\ space$ F

$$u : \Omega \to F$$

We are going to talk more about these in the next section.

1.3 Limitations of Imaging

Imperfections in the manufacturing of the optical components in imaging systems give rise to limitations in image resolution. The resolution of an optical imaging system is diffraction-limited due to the diffraction of light within such systems. ((Hecht, 2002))

Similar to how imperfections in the optical components introduce limitations in resolution, electronic noise produced by the image sensor and film grain introduced by the presence of metallic particles present on the imaging surface can introduce image noise. Image noise is easily recognized as a grainy effect on the photographed image, which is easily seen when photos are taken with maximum magnification with digital cameras or cell phone cameras.

2 Introduction

2.1 What is an image?

Of course, everyone knows what does an image look like. According to the Oxford Dictionary, an image is a picture, photograph, or statue representing somebody/something. To better understand the principle of image denoising, we will focus on introducing different types of images here. Below is prepared by ((Bredies and Lorenz, 2018, Chap 1))

Photography: Photography is the projection of a real-world scene onto a two-$dimensional$ image plane with specific optical devices. The optical elements are focused on a particular plane, called the focal plane, and the farther the object is from the focal plane, the more blurred they appear. Therefore, photos usually have both sharp and blurred areas. Technically, photography is based on chemical reactions that map different values of brightness and color onto film. Each chemical reaction will have some slight uncontrollable changes, so the photographic photos do not entirely match the actual brightness and color values. In particular, the film has a certain $granularity$, which is equal to a certain $noise$ in the picture.

Today, most photos are obtained digitally. Here, brightness and color are measured digitally at certain locations (pixels or picture elements). This results in a matrix of brightness or color values. The digital picture acquisition process can also cause some noise in the picture.

Scans: To digitize photos, one can use a scanner. The scanner illuminates the photo row by row and measures the brightness or color along the line. Typically, this does not cause some additional blur. The scanner operates at a certain resolution, which leads to a $reduction\ in\ information$. Besides, the scanning process may generate some other artifacts. The earlier scans are usually pale and may contain some contaminants—correction of such errors is an essential issue in image processing.

Microscopy: Microscopy is like a combination of photography and scans. One uses measurement technology similar to photography, but since the light cone is very vast, the depth of field is very low. Therefore, objects that are not in the focal plane look very blurry and are invisible. This results in the image being almost $almost\ two-dimensional$.

Indirect imaging: In some cases, image data cannot be directly measured. Prominent examples are computed tomography (CT) and ultrasound imaging. Take CT as an example; the $three$-$dimensional$ image of an object's density is reconstructed by acquiring X-ray scans of the object from different directions. Artifacts are often generated during the process, and due to noise from the X-ray measurement process, the reconstructed image also has $noise$.

Generalized image: Data different from the above can also be regarded as images. For example, in industrial engineering, individual workpieces' smoothness can be checked by measuring the surface. This results in a two-dimensional "elevation profile" that can be treated as an image. In other cases, the chemical concentration or magnetization may have been measured. A slightly different example is liquid chromatography-mass spectrometry (LC/MS). Here, a one-dimensional mass spectrum that measures time correlation. The obtained two-dimensional data has quality and time dimensions and can be processed by imaging technology.

2.2 From image domain to color space

In section **1.2**, we have discussed about image u as a mapping from $image\ domain\ \Omega$ to some $color\ space\ F$, here we will give some examples of each one:

2.2.1 Image Domain

Discrete image domain for n-dimensional images:

$$\Omega = \{1, \ldots, N_1\} \times \cdots \times \{1, \ldots, N_n\}$$

Continuous image domain for n-dimensional images:

$$\Omega = [0, a_1] \times \cdots \times [0, a_n] \quad or \quad \Omega \subset \mathbb{R}^n$$

2.2.2 Color Space

Black-and-white images:

$$F = \{0, 1\}$$

Grayscale images with discrete color space with k-bit depth:

$$F = \{0, \ldots, 2^k - 1\}$$

Color images with k-bit depth for each of N color channels:

$$F = \{0, \ldots, 2^k - 1\}^N$$

Images with continuous grayscale values:

$$F = [0, 1] \quad or \quad F = \mathbb{R}$$

Images with continuous colors:

$$F = [0, 1]^3 \quad or \quad F = \mathbb{R}^3$$

Since the method for image processing is motivated by continuous analysis. We will be focusing on deriving the method for continuous images with continuous color space. Moreover, we will be focusing on the continuous grayscale images.($\Omega \subset \mathbb{R}^2$ and $F = \mathbb{R}$).

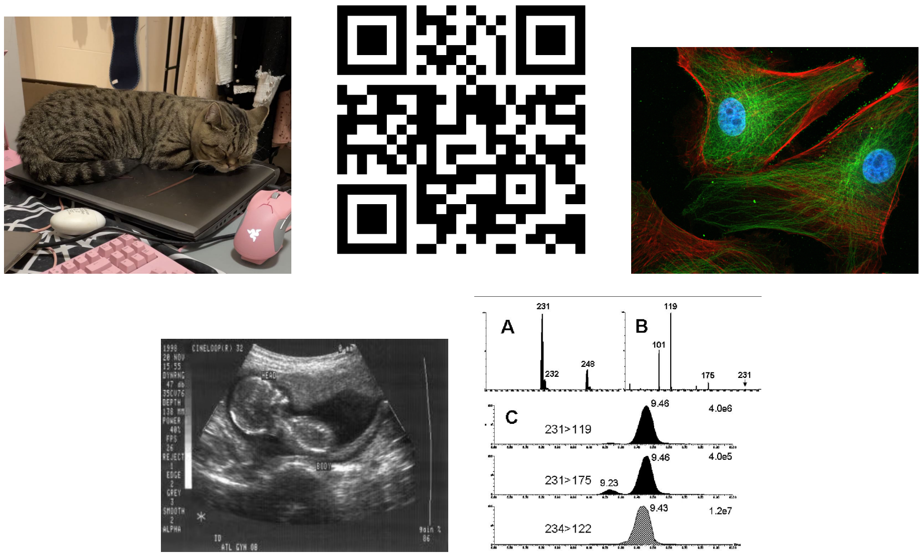


Figure: The figures above show different types of images. The first row starting from the left to the right is A colored photo of a cat, a QR code scan and a Neuro Microscopy image. Second row: left is a medical ultrasound of a baby, right is a liquid chromatography-mass spectrometry.

3 Noise

Have you ever tried to take a photo at night or in any low light conditions? You might observe much more grain or even splotches of discoloration on the pictures taken at night compared to photos taken in a high-light environment. We usually describe this visual distortion as $noise$.

3.1 Mathematical Definition of Image Noise

Consider η as the unknown noise function that we wants to remove from original data of image u^0, let u represents the reconstructed image, then we will get the mathematical expression of the relationship between noise and data from the image:

$$u^0 = u + \eta$$

3.2 Motivation for denoising

Since we do not know the noise η, we need to make assumptions about u and η based on our observations. And we need an expression that measures how "well" a function "fits" the modeling assumption.

Assume there is a real-valued function Φ and a real-value function Ψ, such that

•Φ provides the "Size" of every noise function η.

•Ψ demonstrates how much an image u looks like a 'natural image.' This function uses 'unnatural' for scale 'large' and 'natural' for scale 'small.'

For suitable functions Φ and Ψ, we choose a weight $\lambda > 0$, and for every image u and consequently every noise

$\eta = u^0 - u$:

$$\Phi(u^0 - u) + \lambda\Psi(u)$$

The above expression gives a value of how well both requirements are fulfilled since the 'smaller' the noise, the 'better' the image quality. The case is now reduced to the most familiar problem in this course: finding the optimal control u or finding the minimizing function u.

$$\Phi(u^0 - u^*) + \lambda\Psi(u^*) = \min_u \Phi(u^0 - u) + \lambda\Psi(u)$$

4 Total Variation Denosing

At this stage, we can formally define our problem: Let a noisy image $g(x)$ be

$$g(x) = u(x) + n(x) \tag{1}$$

where $u(x)$ is the original image we want to reconstruct and $n(x)$ is some noise, $x \in \Omega \subset \mathbb{R}^2$ (because an image is usually 2-dimension). Tikhonov regularization shows that the minimization problem we are looking to solve corresponding to the regularizer $F(u)$ is

$$\min_u \lambda F(u) + \frac{1}{2}\int_\Omega |u(x) - g(x)|^2 dx \tag{MAP}$$

It is intuitively correct because a restored image should have a small difference from the original image, but the question is, what are good choices of $F(u)$?

4.1 Choice of F(u)

In the lecture notes prepared by Chambolle et al Chambolle et al. ((2010)) , two regularizers that are usually considered are

$$F_1(u) = \frac{1}{2}\int_\Omega u^2 dx$$
$$F_2(u) = \frac{1}{2}\int_\Omega |\nabla u|^2 dx = \frac{1}{2}\int_\Omega u_{x_1}^2 + u_{x_2}^2 dx_1 dx_2$$

Notice that if we denote $u_{i,j}$ as the pixel at the position (i, j), $u_{x_1}^2 + u_{x_2}^2$ can be written as

$$u_{x_1}^2 + u_{x_2}^2 = \sum_{i,j} |u_{i+1,j} - u_{i,j}|^2 + |u_{i,j+1} - u_{i,j}|^2$$

And a more intuitive way to think about $u_{x_1}^2 + u_{x_2}^2$ is the summation of the difference of conjoint pixels. It measures the complexity of an image with respect to its variation. Since $F_1(u)$ and $F_2(u)$ are convex, thus the Euler-Lagrange equations for the ROF problem will give us an unique value of u . For $F_1(u)$, the Euler-Lagrange equation is

$$\lambda u + u - g = 0$$

And for $F_2(u)$, the Euler-Lagrange equation is

$$\lambda\Delta u + u - g = 0 = \lambda(u_{x_1}^2 + u_{x_2}^2) + u - g = 0$$

The advantage of these choices is that the corresponding Euler-Lagrange equations are linear, and we can derive u easily. Figure below shows the performance of F_1 and F_2

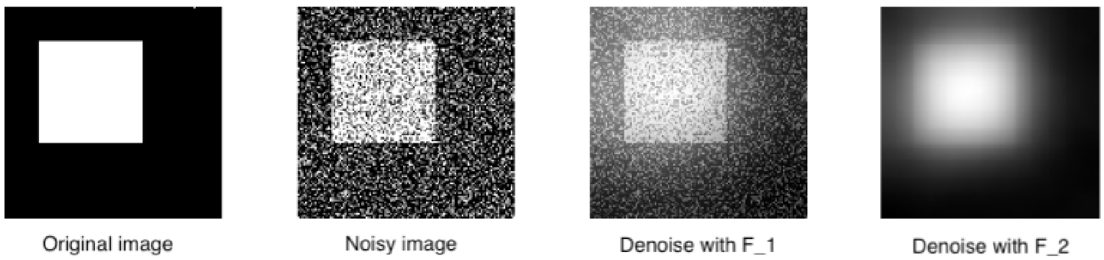


Figure 1: Denosing using F_1 and F_2, image from the lecture notes prepared by Chambolle et al. Chambolle et al. ((2010))

The image from $F_1(u)$ still have large noise; this is because minimize $\frac{1}{2}\int_\Omega u^2 dx$ does not reduce the difference of conjoint pixels; we do not have regularization in this case. In the second case, because of the gradient's square, there is too much regularization that makes the edges disappear. Therefore, both $F_1(u)$ and $F_2(u)$ are not reasonable choices. We need a regularizer F that can make some regularization and preserve the edges simultaneously.

4.2 ROE's Total Variation

In 1992, Rudin, Osher, and Fatemi published a celebrated paper Aubert and Kornprobst ((2006)) that shows how a total variation regularizer can meet such requirements. In the paper, they proposed to choose

$$F(u) = J(u) = \int_\Omega |\nabla u(x)|_2 dx = \int_\Omega \sqrt{u_{x_1}^2 + u_{x_2}^2} dx_1 dx_2 \qquad \text{(ROE's Total Variation)}$$

where $|\cdot|$ denotes the L_2 norm. According to Rudin et al Aubert and Kornprobst ((2006)), the L_1 norm is usually avoided because $\int_\Omega |u|_1 dx$ will produce singularities which cannot be handled algebraically. If you take a close look at ROE's Total Variation, you will find that it is similar to F_2 we discuss before, ROE's Total Variation has a square root while F_2 does not. Indeed, with the same constraints, L_2 is better than L_2^2 most of the time. Here is a demonstration that shows the difference between L_2 and L_2^2

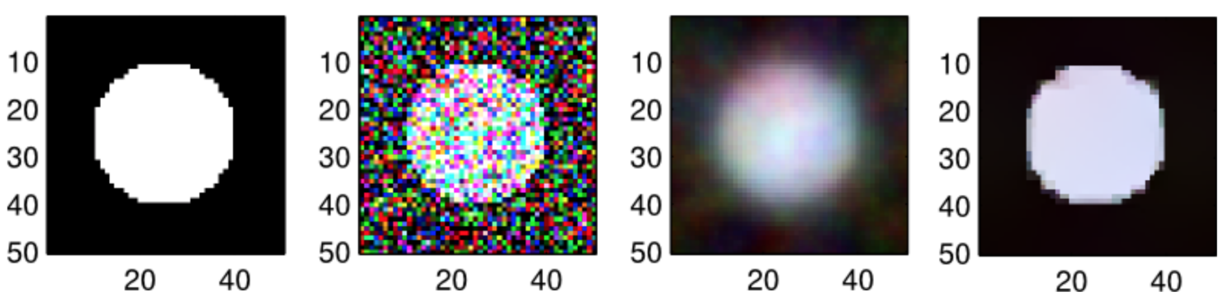


Figure 2: From left to right, the sub-figures are original image, noisy image, denoising using L_2^2 and denoising using L_2 , figure taken from *remi.flamary.com*

We can see that L_2^2 denoising leads to a smooth image rather than a detailed one and the edges are not preserved, but L_2 denoising can keep the sharp edges. The reason behind this is that if we choose $F(u) = \int_\Omega |\nabla u(x)|_2$ as a regularzier, solving the minimization problem will give a discontinuous solution rather than a continuous one, and those discontinuities allow the images to have edges. This the motivation of ROE's Total Variation. Unfortunately, the math behind this is beyond our ability and we can not show it here.

4.3 Convexity

Clearly, the ROE's Total Variation is convex from its definition, we want to show the minimization problem MAP is also convex. The following proof is taken from the lecture notes prepared by Chambolle et al Chambolle et al. ((2010)). Define

$$\mathcal{E}(u) := \lambda J(u) + \frac{1}{2}\int_\Omega |u(x) - g(x)|^2 dx$$

Use the definition of convexity, for any $u_1, u_2 \in \Omega$ and $t \in [0, 1]$, we have

$$J(tu_1 + (1-t)u_2) \leq tJ(u_1) + (1-t)J(u_2)$$

If u and u' are two solution of MAP, then

$$\begin{aligned}\mathcal{E}\left(\frac{u+u'}{2}\right) &\leq \frac{\lambda}{2}(J(\lambda) + J(\lambda')) + \int_\Omega \left|\frac{u+u'}{2} - g\right|^2 \\ &= \frac{1}{2}(\mathcal{E}(u) + \mathcal{E}(u')) - \frac{1}{4}\int_\Omega (u-u')^2 dx \\ &\leq \inf \mathcal{E}\end{aligned}$$

The equality only holds when $u = u'$. Thus $\mathcal{E}$ is convex and the minimizer of $\mathcal{E}$ exists and is unique.

4.4 Euler-Lagrange Equation

There are many methods to solve the MAP, but in our case, we will only talk about the method using the Euler-Lagrange Equation. Formally, the Euler-Lagrange Equation of MAP is given by

$$\lambda \text{ div } \frac{Du}{|Du|} + u - g = 0$$

Expand div $\frac{Du}{|Du|}$ gives us

$$\lambda \frac{\partial}{\partial x_1}\left(\frac{u_{x_1}}{\sqrt{u_{x_1} + u_{x_2}}}\right) + \lambda \frac{\partial}{\partial x_2}\left(\frac{u_{x_2}}{\sqrt{u_{x_1} + u_{x_2}}}\right) + u - g = 0$$

In the paper written by Rudin et al., they also introduced a fast numerical method to solve the above Partial Differential Equation and compute a 'good' local minimum in a very short time.

5 Implementation

5.1 Algorithm

The total variational method for image denoising can be implemented computationally to denoise grayscale images. First, a computational algorithm must be designed from the mathematical algorithm. For this, Chambolle's 2004 paper Chambolle ((2004)) will be used.

The main principle that makes denoising with total variation possible is that signals with excessive and possibly spurious detail have high total variation. Rudin et al. ((1992)) In other words, the integral of the absolute gradient of the signal is high.

Note that the total variation of an image u in discrete form is

$$J(u) = \sum_{1 \leq i,j \leq N} |(\nabla u)_{i,j}|$$

where $i, j = 1, ..., N$.

Using this principle, we can reduce the image noise by reducing the signal's total variation while attempting to preserve details such as edges. Chambolle proposed that by solving

$$\min_{u \in X} ||u - g|| + 2\lambda J(u)$$

given $g \in X$ and $\lambda > 0$, an image will be denoised according to the total variation principle. Chambolle ((2004))

The Euler equation for the above expression is Chambolle ((2004))

$$u - g + \lambda \partial J(u) \ni 0$$

The solution u to the given problem is then given by

$$u = g - \pi_{\lambda K}(g)$$

where $\pi_{\lambda K}$ is the nonlinear projection.

To compute the nonlinear projector $\pi_{\lambda K}(g)$, the following problem must be solved:

$$\begin{aligned} \min\{||\lambda \nabla p - g||^2 : p \in Y, \\ |p_{i,j}|^2 - 1 \leq 0 \forall i, j = 1, ..., N\} \end{aligned}$$

5.2 Code

Here, the translation of the mathematical algorithm to code is presented.

The denoising code is written in python as a function and has a required parameter, an image that is a 2D array of grayscale pictures. This can be easily loaded from a jpeg or png image with

```
f = np.array(plt.imread('image.png'), dtype=float)
```

There are three optional parameters weight, eps, and n_iter . The weight is a float that represents denoising weight. The larger the value of weight, the more denoising the end result will have. The tau parameter represents the time step in each iteration which must be less than or equal to 0.125. Chambolle ((2004)) The eps parameter is a float that represents the relative difference of the value of the cost function. This value is used to terminate the function when the following condition is met:

$$E_{n-1} - E_n < eps \cdot E_0$$

The n_iter is an integer that sets the hard limit on the maximum number of iterations used for the optimization.

The first step of the program is to initialize the variables which will be used later on.

```
def denoise(image, weight=0.1, tau=0.125, eps=2.e-4, n_iter=200):
    n, m = image.shape[0], image.shape[1]
    u = np.zeros((n, m))
    gx = np.zeros((n, m))
    gy = np.zeros((n, m))
    i = 0
```

First, the dimensions of the image array are stored into n and m as integers. Then, three arrays of zeros are created with the same dimension as the input image and are stored as u, gx, and gy. Here, u will be the initial guess to the clean image. gx and gy are the components of variable g, and i is the iteration counter initially set to zero.

The function will initially be setup to iterate n_iter amount of times.

```
    while i < n_iter:
        u_i = u
        uy, ux = gradient(u)
```

At each iteration, the old image array u will be stored as u_i. Then, the gradient of u in the x and y direction is computed with the following function:

```
def gradient(A):
    out = np.zeros((2, ) + A.shape)

    slices1 = [0, slice(0, -1), slice(None)]
    slices2 = [1, slice(None), slice(0, -1)]
    out[tuple(slices1)] = np.diff(A, axis=0)
    out[tuple(slices2)] = np.diff(A, axis=1)
    return out
```

Then, the components of g are calculated using the gradient of u, which was calculated earlier.

```
        gx_new = gx + (tau / weight) * ux
        gy_new = gy + (tau / weight) * uy
```

Next step is to normalize:

```
        norm_coeff = np.maximum(1, np.sqrt(gx_new **2 + gy_new ** 2))
        gx = gx_new / norm_coeff
        gy = gy_new / norm_coeff
```

After normalization, the divergence of g is computed using numpy.rolls:

```
        div_g = (gx - np.roll(gx, 1, axis=1)) + (gy - np.roll(gy, 1, axis=0))
```

Then, the image array u is updated with new values:

```
        u = image + weight * div_g
```

The last step in the iteration loop is to check the stop condition outlined before as $E_{n-1} - E_n < eps \cdot E_0$:

```
        E = np.linalg.norm(u-u_i) / np.sqrt(n*m);
        if i == 0:
            E_init = E
            E_previous = E
        else:
            if np.abs(E_previous - E) < eps * E_init:
```

```
            break
        else:
            E_previous = E
    i += 1
```

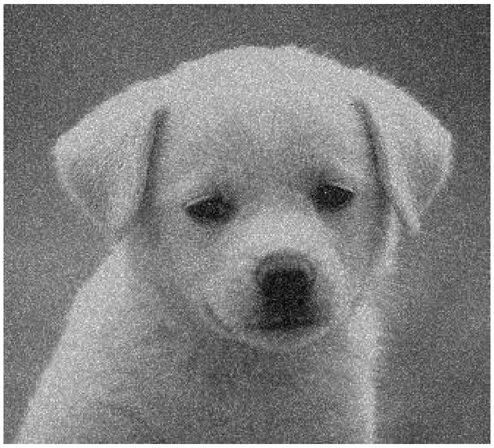
Finally, once the loop finishes, the updated image array u is returned.

* A Jupyter notebook is avaiable for download at this Github repository.

6 Results

An image obtained from the web will be used for denoising. Before denoising, noising will be added with a random number generator changing the image array's grayscale values slightly.

Original Image



Noise Added

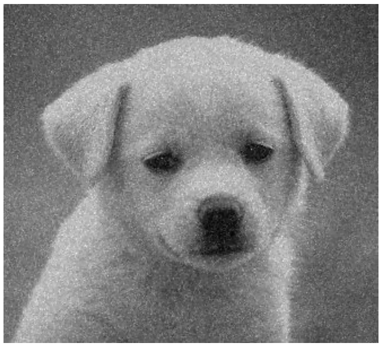
With the default parameter values of weight=0.1, tau=0.125, eps=2.e−4, n_iter =200, the denoised image output is the following:

Denoised Image with Default Parameters

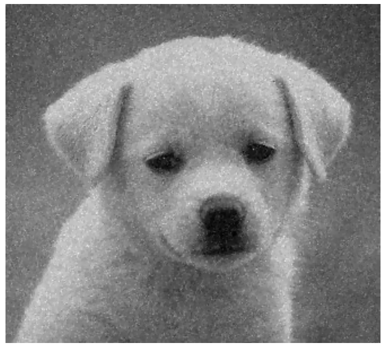
The output image has reduced noise, but this is achieved at the expense of image quality.

6.1 Varying Weight Parameter

An increase in the weight parameter value will result in an image with better denoising. However, the image quality will worsen as the value increases.



(a) weight=0.01



(b) weight=0.02

(c) weight=0.05

(d) weight=0.07

(e) weight=0.1 (default)

Denoised Images with Varying Weight Values

From the five different weight values, it can be observed that as the weight value increases, the noise reduction gets better. However, the image becomes more blurry as the weight value is increased.

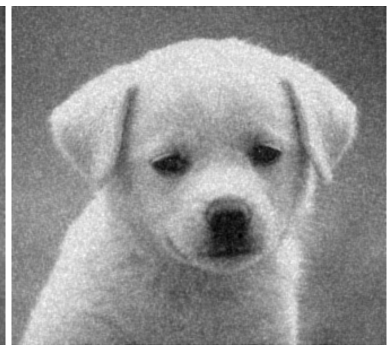
6.2 Varying eps Parameter

(a) eps=2e-4 (default)

(b) eps=4e-4

(c) eps=2e-3

(d) eps=2e-2

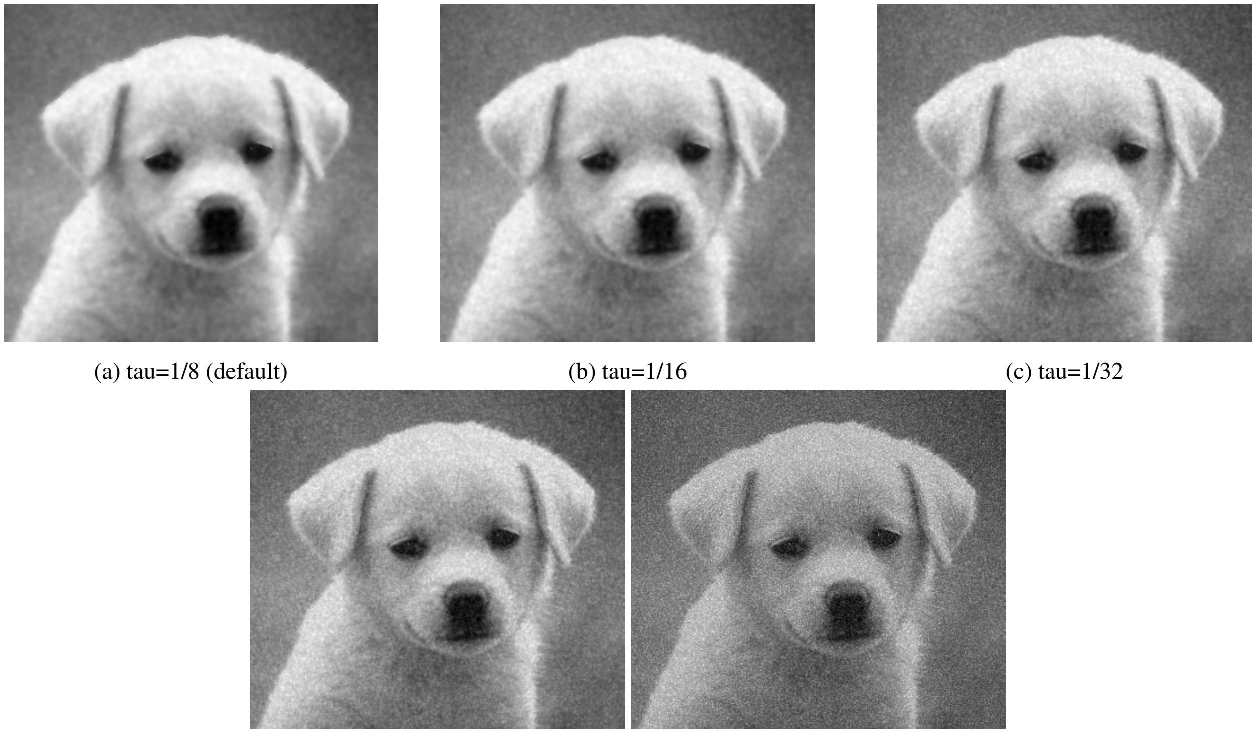


(e) eps=2e-1

Denoised Images with Varying eps Values

Varying the eps parameter also exhibits a noticeable change in the output results. As the eps value increases, the noise reduction decreases. This trend is due to the error check condition $E_{n-1} - E_n < eps \cdot E_0$. A large eps value will terminate the iterative loop quicker hence the decreased denoising effect.

6.3 Varying tau Parameter



(a) tau=1/8 (default) (b) tau=1/16 (c) tau=1/32

(d) tau=1/64 (e) tau=1/128

Denoised Images with Varying tau Values

The effect of tau values is also noticeable. As tau decreases from its max value of $\frac{1}{8}$, the noise reduction decreases. This trend arises from the fact that tau represents the step size in the iterative loop for calculating the value of g, which affects the value of the divergence of g. Although it may be intuitive to increase the n_iter parameter to circumvent the decreased noise reduction effect, doing so does not yield any change due to the error check condition. For all of the image denoising examples, the actual number of loop iterations did not get close to the default maximum of 200 due to the error check condition triggering the loop's breakpoint.

7 Conclusion

In denoising, the first thing to consider is choosing an appropriate regularizer $F(u)$. In Section 4.2, we introduced the problem that if an unwanted regularizer is selected, the reconstructed image may have too little noise reduction or blurry edges compare to the original image. The occurrence of ROE's Total Variation has provided a reasonable solution to this problem. This regularizer perfectly addresses the two issues mentioned above by giving a non-continuous solution u. Moreover, after examining its convexity, the expression is convex across the entire image domain, ensuring that the solution is always a minimizer and unique.

After determined the expressions of $F(u)$ and $J(u)$, in the implementation part, we focused on how to implement the algorithm to get the minimum solution and how the change three parameters: λ(weight), τ(n_iter) and stop ϵ(eps) affect the reconstructed image. In the comparison, it is clear that the effects of two weight parameters λ and τ on image denoising are similar: the larger two parameters produce more blurry images with less noise. The stopping parameter ϵ, in contrast, produces a less blurry image with more noise when it gets larger.

References

Dither a grayscale image. `https://codegolf.stackexchange.com/questions/26554/dither-a-grayscale-image/26574`, May 2014.

G. Aubert and P. Kornprobst. *Mathematical problems in image processing: partial differential equations and the calculus of variations*, volume 147. Springer Science & Business Media, 2006.

K. Bredies and D. Lorenz. *Mathematical Image Processing*. Springer, 2018.

A. Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20(1):89–97, Jan 2004. ISSN 1573-7683. doi: 10.1023/B:JMIV.0000011325.36760.1e. URL `https://doi.org/10.1023/B:JMIV.0000011325.36760.1e`.

A. Chambolle, V. Caselles, D. Cremers, M. Novaga, and T. Pock. An introduction to total variation for image analysis. *Theoretical foundations and numerical methods for sparse recovery*, 9(263-340):227, 2010.

E. Hecht. *Optics*. Pearson education. Addison-Wesley, 2002. ISBN 9780805385663. URL `https://books.google.ca/books?id=7aG6QgAACAAJ`.

L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.