TP- IMA 203 Variational methods

Giovanni Benedetti da Rosa

January 22, 2024

1 Denoising with quadratic regularization

1.1

$$E_1(u) = \|u - v\|^2 + \lambda \|\nabla u\|^2$$

$$E_1(u) = |\delta * u - v\|^2 + \left\|\sqrt{\lambda} K_x * u\right\|^2 + \left\|\sqrt{\lambda} K_y * u\right\|^2 = \sum_{i=1}^{3} \|K_i * u - v_i\|^2$$

Where K_y is the vertical derivative filter and K_x is the horizontal derivative filter. Using the Fourier transform, we obtain:

$$\hat{E}(f) = \sum_{i=1}^{3} \sum_{\omega} \left|\hat{K}_i * \hat{u} - \hat{v}_i\right|^2 = \sum_{\omega} \sum_{i=1}^{3} \left|\hat{K}_i * \hat{u} - \hat{v}_i\right|^2$$

So for each frequency(i) we need to minimize a problem of the form :

$$|\alpha t - \beta|^2$$

We can find that the solution to this is:

$$\hat{u} = \frac{\sum_i \overline{\hat{K}_i(\omega)} \hat{v}_i}{\sum_i \left|\overline{\hat{K}_i(\omega)}\right|^2}$$

We can denoise the image using the function "minimisation_quadratique(image,lamb)", where λ is the regularization term in E_1.

This function the expression showed above is implemented in the function "resoud_quad_fourier(K,V)".

This function is called by "minimisation_quadratique(v,lamb)", which creates the arrays K and V, that are in our case $K = [\sqrt{\lambda} K_x, \sqrt{\lambda} K_y, \delta]$ and $V = [0, 0, v]$.


Figure 1: $\lambda = 10^{-3}$ and $\lambda = 10^{6}$

1.2

If we increase a lot the value of λ, the energy tends to be represented mainly by regularization term.

$$E_1(u) \underset{\lambda \to \infty}{\longrightarrow} \lambda \|\nabla u\|^2$$

In this case minimize the image would becomes mainly minimizes the gradient. As the gradient of an image becomes smaller, more constant the image is the image start to gets blurry.

On the otherhand if we reduce a lot the λ value, the energy is mostly the data attachment term.

$$E_1(u) \underset{\lambda \to 0}{\longrightarrow} \|u - v\|^2$$

And we still get a noise image as an output.

1.3

In order to find the parameter for which the constructed image u is at the same distance from the degraded image v as is the perfect picture, we used the bissection method to find the zero crossing of a function in the form:

$$f(\lambda) = \|\tilde{u}(\lambda) - v\|^2 \|u - v\|^2$$

The result found was $\lambda = 3.21$

Here is the implemented code.

```
1 def bisection_lamb(im, br, lamb_a, lamb_b, epsilon=0.0001, max_iterations=1000):
2     imb = degrade_image(im, br**2)
3     imr_a = minimisation_quadratique(imb, lamb_a)
4     imr_b = minimisation_quadratique(imb, lamb_b)
5     dist_a = np.linalg.norm(im - imb)**2 - np.linalg.norm(imr_a - imb)**2
6     dist_b = np.linalg.norm(im - imb)**2 - np.linalg.norm(imr_b - imb)**2
7
```

```
    if dist_a * dist_b > 0:
        raise ValueError("Initial guesses do not bracket a root (dist_a and dist_b have the

    iteration = 0
    while iteration < max_iterations:
        lamb_c = (lamb_a + lamb_b) / 2
        imr_c = minimisation_quadratique(imb, lamb_c)
        dist_c = np.linalg.norm(im - imb)**2 - np.linalg.norm(imr_c - imb)**2

        if np.abs(dist_c) < epsilon or (lamb_b - lamb_a) < epsilon:
            return lamb_c

        if dist_c * dist_a > 0:
            lamb_a = lamb_c
            dist_a = dist_c
        else:
            lamb_b = lamb_c
            dist_b = dist_c

        iteration += 1

    raise Exception('Maximum iterations reached without convergence')

bisseclamb = bisection_lamb(im, 5, 0.1, 5)
print(bisseclamb)
```

1.4

Now we iterate over ange of values to find the one that minimizes the expression

$$f(\lambda) = \|\tilde{u}(\lambda) - v\|^2$$

The code provided below has found the value $\lambda = 1.18$, that minimizes the expression.

```
error_best=norm2(imb)+10
for k in np.arange(0.1,3,0.01) :
    lamb=k
    res=minimisation_quadratique(imb,lamb)
    error=norm2(im-res)
    if error<error_best:
        error_best=error
        lamb_best=lamb
        res_best=res
print(lamb_best)
```

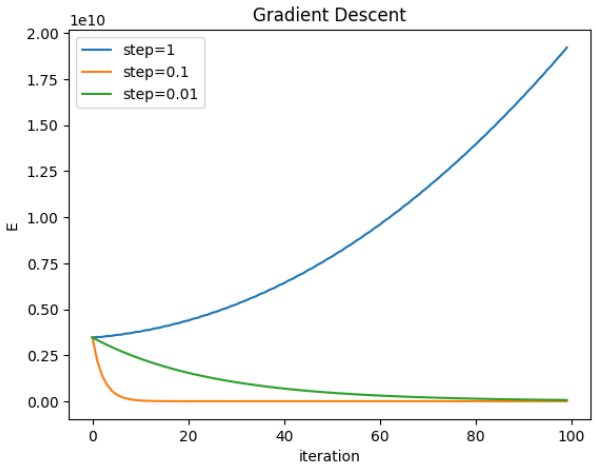
2 Denoising with total variation(VT)

2.1 Gradient descent

Now we want minimize the following energy expression:

$$E_2(u) = \|u - v\|^2 + \lambda \|\nabla u\|_1$$

Using gradient descent algorithm, with $\lambda = 1$, we arrive in the following graph.



We can easily check that for a big step size the algorithm does not converge to a minimum, but for smaller values it goes almolst to the same value.

2.2 Chambolle projection

Using the same step size and λ values the Chambole projection(4s) was more than two times faster than the gradient descent(1.5s). Both results are similar.

3 Comparison

In order to better compare both methods, we implement the same method described on question 1.4 to determine the best λ values.

Here is a table that summarizes the results:

Method	λ	Time(s)
Gradient Descent - Quadratic	1.2	0.5
Chambole - VT	41.39	1.5

In the following picture there are the results of the two methods applying the method described before.

Figure 2: Chambolle and Gradient Descent