# Lab 4: ROF model for image denoising

Quim Llimona (146662)

UPF

In this lab, we will implement a better model for image denoising than the one we already implemented in a previous lab, that relies on not squaring the residual to avoid punishing high gradients that arise on the edges.

## 1 Duality

A generic function that will come useful to solve the problem is the follwing, because the energy is a specific case of it:

$$f(x) = \|Ax\|_{\mathbb{R}^m} + \frac{1}{2\lambda}\|x - b\|^2_{\mathbb{R}^n}$$

Where $A$ is a $m \times n$ and $b \in \mathbb{R}^n$. The function is not differentiable when $Ax = 0$, because its derivative involves the absolute value function. However, we can make a trick to make it differentiable: we can write this as the solution of a more general problem, and then solve the general problem in a different way. This general problem relies on a special definition of the norm:

$$\|x\|_{\mathbb{R}^n} = \max_{\|\xi\|_{\mathbb{R}^n} \leq 1} \langle x, \xi \rangle$$

So, we can recover $\|x\|$ by maximizing the expression on the vector $\xi$, with the constraint that its own norm must be less or equal than one. Notice that if this expression was the only definition of norm we had, then this constraint would run into an infinite loop! However, we are using it only as a tool to brigde non-differenciability, so we can just assume that this norm in the constraint is defined as usual.

Before carrying on, let's prove that this definition is correct strarting from:

$$\langle x, \xi \rangle = \|x\|\|\xi\| \cos(\theta)$$

The variables are in the following ranges:

$$0 \leq \xi \leq 1, \quad -1 \leq \cos(\theta) \leq 1$$

If we maximize both separately, we get $\xi = 1$ and $\cos\theta = 1$, which holds when $x$ and $\xi$ are orthogonal. We can do this because the angle $\theta$ depends on $\xi$. Therefore, the definition is valid.

Let $C$ be the feasible for $\xi$, that is, where $\|\xi\| \leq 1$, the generic function can now be rewritten as:

$$f(x) = \max_{\xi \in C} G(x, \xi), \quad \text{where} \quad G(x, \xi) = \langle Ax, \xi \rangle_{\mathbb{R}^m} + \frac{1}{2\lambda} \|x - b\|^2_{\mathbb{R}^n}$$

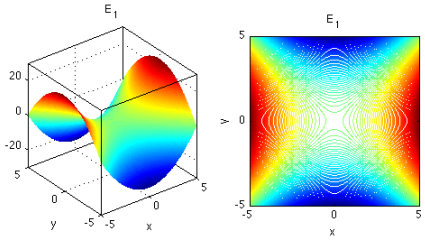
Recall that the goal of all this was to make $f(x)$ differentiable, which is a condition we need to meet in order to minimize it with iterative algorithms. Now we can. The problem can be stated as:

$$\min_x \max_{\xi \in C} G(x, \xi)$$

If the function $G$ is convex on $x$ and concave on $\xi$, then we can swap the min and the max, and solve the min first with a generic $\xi$. The solution will be a saddle point of the function. Before developing on this, let's recall what saddle points and convexity are with some examples.

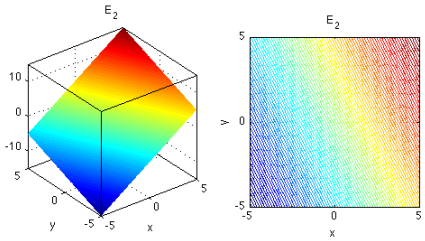
## 1.1 Convexity and saddle points

$$f(x, y) = x^2 - y^2$$



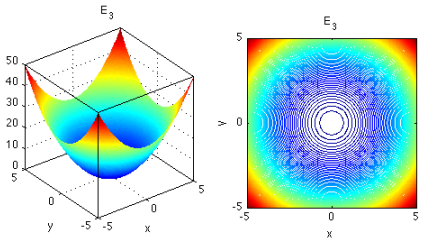

Given a fixed $x_0$, the function $f(y) = x_0^2 - y^2$ is an inverted parabola, which is strictly concave. Given a fixed $y_0$, the function $f(x) = x^2 - y_0^2$ is a parabola, which is strictly convex. Therefore, the function has a saddle point, which is located at $(0, 0)$.

$$f(x, y) = 2x + y$$



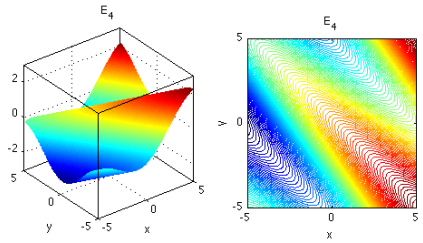

This function is linear on both x and y. Linear functions are both concave, in the sense that when unbounded they don't have any local minima, and convex, in the sense that when unbounded they don't have any local maxima. However, the opposite doesn't hold: being convex doesn't mean having a minimum, and being concave doesn't mean having a maximum. That's because they aren't strictly convex or concave. The function, therefore, hasn't got any saddle point, even being concave in one variable and convex in the other.

$$f(x, y) = x^2 + y^2$$




By fixing any of the two variables, the function becomes a parabola, which is stricly convex. Therefore, the function is convex in the two variable, and has got a single minimum and no maxima when unbounded.

$$f(x, y) = \cos\left(\frac{3}{5}x + y\right) + \frac{x}{4}$$




In this function there is a cosine involved both in the x and the y, so it's neither convex or concave. It doesn't have local saddle points either; to understand why, let's make a change of variables. Let $u = x + y$ and $v = x - y$. The function becomes: $f(u, v) = \cos(3/5u) + 0.25(u + v)$. The function is linear on v, therefore it hasn't got minima or maxima in v. This implies that it can't have saddle points, because a saddle point needs a maximum in one variable and a minimum in the other.

We formulated our problem in a way that would allow solving it with a min-max approach. However, the part we seek to maximize is linear on $\xi$, because the dot product is a linear operator. According to the examples, it doesn't have a maximum! However, that only holds if the function are unbounded. If the function is unbounded, which is the case, it's very likely that the maximum lies on the boundary, altough it won't have a null gradient. We will add a Lagrange multiplier that will make the gradient of the lagrangian become 0 if there is a maximum on the boundary even if the gradient of the original function is non-zero.

The solution to the problem resides on the following obervation: since the function is convex on $x$ (the variable being minimized) and concave on $\xi$ (the variable being maximized), we can swap the max and the min. Then, we can find first the minimum on x with a generic $\xi$ and then maximize it; that is, solve the dual problem. Another option is to swap continously between min and max and solve the two of them at the same time by finding directly a saddle point. Let's explore the first option first.

## 1.2 Primal-dual approach

We can solve both problems at the same time by making a gradient descent on $x$ and a gradient ascent on $\xi$ at each iteration. The only thing to bear in mind is that $\xi$ is constrained. Projecting the new $\xi$ onto the feasible set at each iteration is enough, that is, we will keep its direction and set its length to a maximum of 1 with the following projector:

$$P_C(\nu) = \frac{\nu}{\max(1, \|\nu\|_{\mathbb{R}^n})}$$

We need the gradient on x, $\nabla_x G(x, \xi)$, in order to make the gradient descent. The function $G$ can be rewritten using only dot products as:

$$\begin{aligned} G(x,\xi) &= \langle Ax, \xi\rangle + \frac{1}{2\lambda}\langle x-b, x-b\rangle \\ &= \langle Ax, \xi\rangle + \frac{1}{2\lambda}\langle x, x\rangle - \frac{1}{\lambda}\langle x, b\rangle + \frac{1}{2\lambda}\langle b, b\rangle \\ &= \langle x, A^T\xi\rangle + \frac{1}{2\lambda}\langle x, x\rangle - \frac{1}{\lambda}\langle x, b\rangle + \frac{1}{2\lambda}\langle b, b\rangle \end{aligned}$$

Its gradient on x is:

$$\begin{aligned} \nabla_x G(x,\xi) &= A^T\xi + \frac{x}{\lambda} - \frac{b}{\lambda} \\ &= A^T\xi + \frac{1}{\lambda}(x-b) \end{aligned}$$

The gradient on $\xi$ allows the gradient ascent. It can be calculated from the first formulation of $G(x, \xi)$, by observing that the term $\|x-b\|^2$ doesn't depend on $\xi$:

$$\nabla_\xi G(x,\xi) = Ax$$

## 1.3 Dual approach

If the gradient on $\xi$ is difficult to calculate, there is another way to solve the problem: by minimizing the energy on $x$ with a generic $\xi$, then maximizing $\xi$ on the minimized energy (that doesn't involve $x$) and finally putting the $\xi$ that gives the maximum into the expression we found for $x$ when minimizing on it. Let's see how this works in our problem.

The -unconstrained- minimum on $x$ can be found by setting the gradient of $G$ on $x$, that we already calculated, to 0. The resulting point will depend on $\xi$; let's call it $x^*(\xi)$:

$$\begin{aligned} \nabla_x G(x,\xi) &= 0 \\ A^T\xi + \frac{1}{\lambda}\left(x^*(\xi) - b\right) &= 0 \\ \frac{x^*(\xi)}{\lambda} &= \frac{b}{\lambda} - A^T\xi \\ x^*(\xi) &= b - \lambda A^T\xi \end{aligned}$$

Now we can put this into the original function. The dependence on $x$ will be removed and we will get the dual function:

$$f_D(\xi) = \min_x G(x,\xi) = G\left(x^*(\xi),\xi\right) = \langle Ab,\xi\rangle - \frac{\lambda}{2}\|A^T\xi\|^2$$

To solve the problem, we need to maximize this function with the constraint $\xi \in C$. We can do it with a projected gradient ascent as described before, if we know its gradient. The dual function can be rewritten using only scalar products as:

$$\begin{aligned} f_D(\xi) &= \langle Ab,\xi\rangle - \frac{\lambda}{2}\|A^T\xi\|^2 \\ &= \langle Ab,\xi\rangle - \frac{\lambda}{2}\langle A^T\xi, A^T\xi\rangle \\ &= \langle Ab,\xi\rangle - \frac{\lambda}{2}\langle AA^T\xi,\xi\rangle \end{aligned}$$

Therefore, its gradient is given by:

$$\nabla f_D(\xi) = Ab - \lambda AA^T\xi$$

Given the dual optimum $\xi^*$ that maximizes $f_D$, the primal optimum (which is the number we really care about) can be recovered by substituting $\xi^*$ into $x^* = x^*(\xi^*)$:

$$x^* = b - \lambda A^T\xi^*$$

# 2 The ROF model for image denoising

## 2.1 Formulation of the energy

In a previous lab we made an image denoising application that relied on a quadratic energy. Its problem was that it actually smoothed the entire image, and an edge finder was needed to prevent the pixels on the edges from being processed. This can be avoided by not squaring the gradient of the image in the energy:

$$E(u) = \sum_{i=1}^{N}\sum_{j=1}^{M} |\nabla^+ u_{ij}| + \frac{1}{2\lambda}\sum_{i=1}^{N}\sum_{j=1}^{M}(u_{ij} - f_{ij})^2$$

The problem is analogous to the toy function we just minimized. The non-differentiability of the norm can be avoided by introducing an auxiliary variable to be maximized. However, we are not taking the gradient of a single vector but the sum of the gradients of each pixel on the image. Thus, we need another vector-valued image with all the vectors $\xi$ that will be maximized:

$$\min E(u) = \min_u \max_{\xi \in C} \sum_{i=1}^{N}\sum_{j=1}^{M} \langle \nabla^+ u_{ij}, \xi_{ij} \rangle + \frac{1}{2\lambda}\sum_{i=1}^{N}\sum_{j=1}^{M}(u_{ij} - f_{ij})^2$$

The feasible set for $\xi$ is modified as well; we will require that each one of the vectors $\xi_{ij}$ has a norm not larger than 1. Let $h = P_C(g)$ be the projector, then:

$$h_{ij} = \frac{g_{ij}}{\max(1, |g_{ij}|)} = \frac{(g_{1,ij}, g_{2,ij})}{\max\left(1, \sqrt{g_{1,ij}^2 + g_{2,ij}^2}\right)}$$

Recall that the transpose of the forward gradient, $\nabla^+$, is given by the negative backward divergence, $-\operatorname{div}^-$. Then, using the formulas that we calculated before the gradients of the energy are:

$$\nabla_\xi G(u, \xi) = \nabla^+ u$$

$$\nabla_u G(u, \xi) = -\operatorname{div}^- \xi + \frac{1}{\lambda}(u, f)$$

This is sufficient to implement a primal-dual solution of the problem, by making a gradient descent on $u$ and a projected gradient ascent on $\xi$ using the projector defined above. The dual problem, which we already saw that can be solved faster, needs the dual energy and its gradient. As we saw before it is given by:

$$E_D(\xi) = \langle \nabla^+ f, \xi \rangle - \frac{\lambda}{2}\|div^- \xi\|^2$$

$$\nabla E_D(\xi) = \nabla^+ f + \lambda \nabla^+ \operatorname{div}^- \xi = \nabla^+(f + \lambda \operatorname{div}^- \xi)$$

We can find the optimal $\xi^*$ by maximizing the energy with a projected gradient ascent. Then, the optimal $u^*$ is given by:

$$u^*(\xi) = f + \lambda \operatorname{div}^- \xi^*$$

## 2.2 Results

The parameter $\lambda$ controls the attachment to the original image. The higher it is, the less the image looks like the original one. We must find a balance between being different on terms of noise but equal to preserve the objects and textures. This model preserves all the edges without needing some weights from an estimation, which yelds much better resuls. The textures are lost because they get smoothed out with high lambdas, as can be seen on the following examples:

Noisy image

Ground truth

$\lambda = 10$

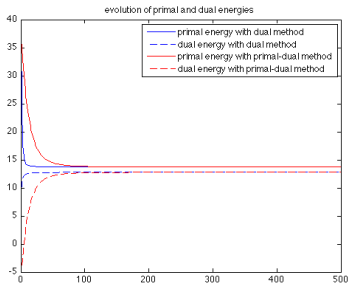
$\lambda = 20$

$\lambda = 30$

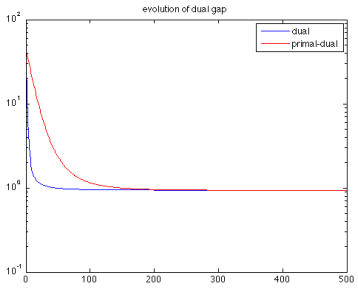
$\lambda = 50$

The best one, according to my own criteria, is the one with $\lambda = 20$.

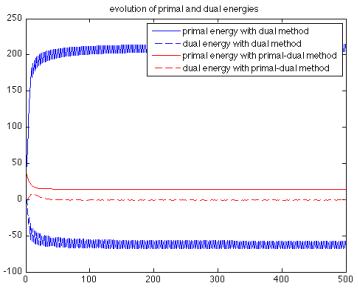
The step sizes, $\delta$ and $\theta$, control how fast the method converges to the minimum. However, if we try to go too fast there is no convergence. The default step size was the best one, because setting it smaller only increases the convergence time. We can see that the dual method is much faster than the primal-dual. We can see as well that the default gap does not converge to 0, but something colse to 1. That is, the primal and dual energies converge to slightly different numbers. It might be due to floating-point precision errors.



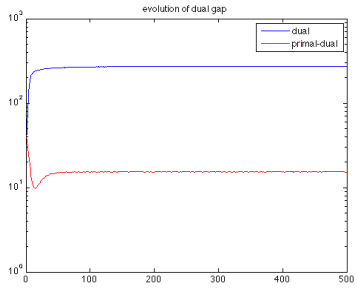

Energy with default step



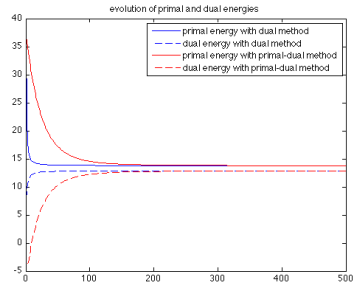

Dual gap with default step



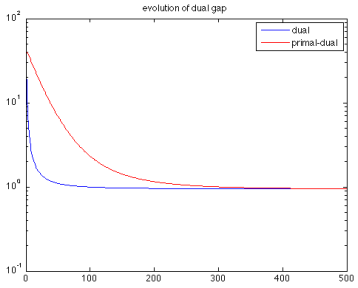

Energy with 2x default step



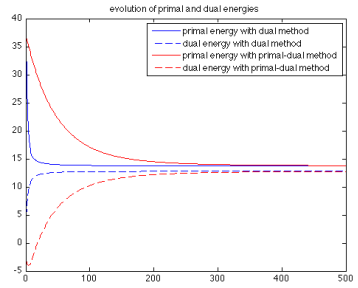

Dual gap with 2x default step



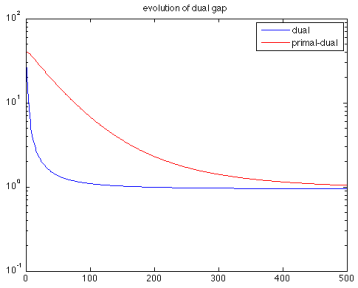

Energy with 0.5x default step




Dual gap with 0.5x default step




Energy with 0.25x default step




Dual gap with 0.25x default step