BME-Paris Master - UE Math 2

Signal and image denoising

In this session we will use the gradient descent algorithm for the optimization of signal denoising functionals. We will try this on synthetic test signals and images.

1D signal denoising

Let $y = (y_1, y_2, \ldots, y_n) \in \mathbb{R}^n$ be a discrete 1D signal, which we suppose to be noisy. We will denoise this signal by minimizing one of the following functionals : we define for every $x \in \mathbb{R}^n$,

$$J_{H^1}(x) = \sum_{i=1}^{n}(x_i - y_i)^2 + \lambda \sum_{i=1}^{n-1}(x_{i+1} - x_i)^2,$$

$$J_{TV_\varepsilon}(x) = \sum_{i=1}^{n}(x_i - y_i)^2 + \lambda \sum_{i=1}^{n-1}\sqrt{\varepsilon^2 + (x_{i+1} - x_i)^2},$$

where $\lambda > 0$ and $\varepsilon > 0$ are fixed parameters. Minimization of these functionals will be performed using the gradient descent algorithm with fixed stepsize.

We first create a synthetic test signal and add noise to it :

```
t = linspace(-pi/2,pi/2,1000);
y_org = cos(t).*sign(t);
```

The array `y_org` now contains values of the function $\cos(t)\,\mathrm{sign}(t)$ in the range $t \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. We will use this as our test clean signal. You can visualize it with

```
plot(y_org)
```

We now add white noise to `y_org` to create our noisy signal `y` :

```
y = y_org + 0.1*randn(size(y_org));
plot(y)
```

To go further we need to write the functions which will compute the values of the functionals and their gradients.

Question 1 Write two functions

```
function J = EvalJH1(x,y,lambda)
function J = EvalJTV(x,y,lambda,eps)
```

which compute the values of $J_{H^1}(x)$ and $J_{TV_\varepsilon}(x)$ for an input vector x.

Question 2 Compute the differentials $DJ_{H^1}(x)$ and $DJ_{TV_\varepsilon}(x)$ and write two functions

```
function G = GradH1(x,y,lambda)
```

`function G = GradTV(x,y,lambda,eps)`
which compute the gradients of the functionals. The output vector G must be of same size as x.

We can now try the gradient descent algorithm and visualize the result.

Question 3 Write a Matlab script which defines `y_org` and `y`, performs $N = 1000$ iterations of the gradient descent algorithm on J_{H^1} with $\lambda = 100$ and stepsize $\eta = 0.001$, and plot the denoised signal `x`. Additionally you can store the sequence of values of the functional at each iteration and plot it in a separate figure. Also try different values for the parameter λ.

Question 4 Now perform the gradient descent on J_{TV_ε} with $N = 1000$, $\lambda = 1$, $\varepsilon = 0.001$ and $\eta = 0.001$. Try different values for λ and η. Compare the two approaches.

Question 5 (*) For the minimization of J_{H^1}, the use of the gradient descent algorithm is in fact not needed, because the solution can be computed directly by solving $\nabla J_{H^1}(x) = 0$. To see this, show first that $J_{H^1}(x)$ can be written in the form $J_{H^1} = \|x - y\|^2 + \lambda\|Ax\|^2$, where A is a $(n-1) \times n$ matrix to be found. Deduce that the equation $\nabla J_{H^1}(x) = 0$ is a linear system which can be written in matrix form $Mx = b$, for some matrix M and vector b. Use Matlab to solve the system and retrieve the solution found by the gradient descent algorithm.

Image denoising

In this part signals will be 2D grayscale images $u = (u_{ij}) \in \mathbb{R}^{m \times n}$, where u_{ij} gives intensity at pixel (i, j). If $v = (v_{ij})$ is the noisy image, we now define :

$$J_{H^1}(u) = \sum_{i=1}^{m}\sum_{j=1}^{n}(u_{ij} - v_{ij})^2 + \lambda\sum_{i=1}^{m-1}\sum_{j=1}^{n-1}(u_{i+1,j} - u_{ij})^2 + (u_{i,j+1} - u_{ij})^2,$$

$$J_{TV_\varepsilon}(v) = \sum_{i=1}^{m}\sum_{j=1}^{n}(u_{ij} - v_{ij})^2 + \lambda\sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\sqrt{\varepsilon^2 + (u_{i+1,j} - u_{ij})^2 + (u_{i,j+1} - u_{ij})^2},$$

Download the two test images : hill and cameraman, and save them to your working folder. We load the first image and display it :

```
v_org = imread('cameraman.png');
imagesc(v_org)
colormap gray
```

`v_org` is a matrix of integers in the range 0 to 256, but we need to work with real values :

```
v_org = double(v_org);
```

Now we can add noise :

```
v = v_org + 20*randn(size(v_org));
imagesc(v)
colormap gray
```

Question 6 Proceed as in the first part : write functions `EvalJH12D`, `EvalJTV2D`, compute the differentials of the functionals, then write functions `GradJH12D`, `GradJTV2D`, and write gradient descent algorithms for the two functionals. You can use the following parameters : $N = 200$, $\lambda = 2$, $\eta = 0.01$ for J_{H^1}, and $N = 200$, $\lambda = 20$, $\eta = 0.01$, $\varepsilon = 0.001$ for J_{TV_ε}. Compare the two approaches.