Coursework 2 for Inverse Problems (MA505250)

This coursework accounts for 30% of your final grade.

The deadline is 3rd April 2020.

Email cmhsp20@bath.ac.uk typed up solutions and your Matlab scripts.

Exercise 1 (Bregman distances). Let $J : \mathcal{U} \to \bar{\mathbb{R}}$ be a convex function. Let $u, v \in \mathcal{U}$ and $p \in \partial J(v)$ be an element of the subdifferential. Then the *Bregman distance* of J at u, v is defined as

$$D_J^p(u, v) := J(u) - J(v) - \langle p, u - v \rangle .$$

In this exercise, we will investigate the properties of the Bregman distance.

1. Show that Bregman distances may not be symmetric, i.e. give an example of a J and $u, v \in \mathcal{U}$ with $p \in \partial J(v), q \in \partial J(u)$ so that
$$D_J^p(u, v) \neq D_J^q(v, u) .$$

2. Show that a vanishing Bregman distance may not imply that the two arguments are already the same, i.e. $D_J^p(u, v) = 0 \not\Rightarrow u = v$? What if J is strictly convex?

[3 marks]

Exercise 2 (Stability bounds). Let $A : L^2(\Omega) \to L^2(\Omega)$ be a bounded linear operator and let $u^\dagger \in BV(\Omega) \cap L^2(\Omega)$, $f_\delta \in L^2(\Omega)$ and $f = u^\dagger \in L^2(\Omega)$ be such that $\|f_\delta - Au^\dagger\|_{L^2} \leqslant \delta$. Consider

$$u_\delta \in \operatorname{argmin}_{u \in L^2(\Omega)} \alpha J(u) + \frac{1}{2}\|Au - f_\delta\|_{L^2}^2,$$

where $J : L^2(\Omega) \to (-\infty, +\infty]$ is a convex functional. Suppose that there exists $p \in L^2(\Omega)$ such that $A^*p \in \partial J(u^\dagger)$. Show that

$$\left\|Au_\delta - Au^\dagger\right\| + |J(u_\delta) - J(u^\dagger)| \leqslant C\delta(\|p\|_{L^2} + 1)$$

provided that $\alpha \sim \delta$.

[3 marks]

Exercise 3 (Total Variation). Consider total variation $\mathrm{TV} : L^1(\Omega) \to \bar{\mathbb{R}}$ as defined in the lecture.

1. Show that TV is convex.
2. Show that TV is lower semi-continuous with respect to L^1 convergence.
3. Show that TV is a seminorm on $BV(\Omega)$ and that $BV(\Omega)$ is a Banach space with norm $\|u\|_{BV} = \mathrm{TV}(u) + \|u\|_{L^1}$.
4. Some special forms of the total variation:
   (a) Let $u \in W^{1,1}(\Omega)$. Prove that in this case $\mathrm{TV}(u) = \int_\Omega |\nabla u|$ where ∇u is the gradient of u in the weak sense.
   (b) Let $\Omega \subset \mathbb{R}^2$, and let $B_r = B(0, r) \subset \Omega$ denote the disc centred at the origin with radius $r > 0$. Define χ_{B_r} the indicator function of B_r and compute its total variation.

[7 marks]

Exercise 4 (Primal-Dual). This exercise is on the use of Primal-Dual iterates

1. Derive an expression for $\mathrm{prox}_{\alpha\|\cdot\|_2}$ for $\alpha > 0$. Hint: consider the convex conjugate of $x \mapsto \|x\|_2$ and apply Moreau's identity.
2. Let $F : x \in \mathbb{R}^n \mapsto \|x\|_1 = \sum_{j=1}^n |x_j|$ and $G : z \in \mathbb{R}^{2n} \mapsto \|z\|_{2,1} = \sum_{j=1}^n \sqrt{z_{j,1}^2 + z_{j,2}^2}$. Derive expressions for $\mathrm{prox}_{\alpha F}$ and $\mathrm{prox}_{\alpha G}$ for $\alpha > 0$.
3. Consider the following settings:
   (a) ($TV + L^1$) Let $u_0 \in \mathbb{R}^n$ and $\varepsilon \in \mathbb{R}^n$ be impulse noise. Consider the observation
   $$f = u_0 + \varepsilon.$$
   One popular way of dealing with sparse noise is to consider the following $TV + L^1$ model
   $$\min_{u \in \mathbb{R}^n} \ \|Du\|_1 + \frac{1}{\lambda}\|u - f\|_1,$$
   where D is the discrete gradient operator and λ is the regularisation parameter.
   (b) (TV deconvolution) Let A be a 2D discrete Gaussian convolution operator (see Exercise 4 of Coursework 1). Consider
   $$\min_{u \in \mathbb{R}^{n \times n}} \ \|Du\|_{2,1} + \frac{1}{2\lambda}\|Au - f\|_2^2$$
   where D is isotropic discrete 2D gradient operator (see appendix A), and $f = Au_0 + \varepsilon$ with u_0 being an given image and ε random Gaussian noise.

   For each of these problems, formulate the primal-dual iterates (and write down this formulation in your typed up report) and provide an implementation in Matlab. In the case of (a), let u_0 be a piecewise constant vector (generate one with at least 1 jump) and ε is chosen to be impulse noise. In the case of (b), let u_0 be the Matlab phantom and $n = 256$, and ε is chosen to be Gaussian noise. Display your reconstruction with different noise levels and different regularisation parameters.

[10 marks]

Exercise 5 (Kalman filter). Consider the discrete model

$$x_0 = 0, \quad x_1 = 1, \quad \text{and for } k = 1, \dots, K, \quad x_{k+1} - 2x_k + x_{k-1} = -\omega^2 x_k.$$

This is a numerical discretisation of the 1D harmonic oscillator:

$$x'' + \Omega^2 x = 0.$$

Define the state vector $u_k = (x_k, x_{k-1})^\top$, and note that $u_k = M u_{k-1}$ where

$$M = \begin{pmatrix} 2 - \omega^2 & -1 \\ 1 & 0 \end{pmatrix}$$

Choose $K = 1000$ and suppose that for every $p = 50$ time units, we observe

$$y_k = H u_k + \xi_k, \quad \text{where} \quad H = (1, 0) \quad \text{and} \quad \xi_k \sim \mathcal{N}(0, r^2).$$

1. Implement the Kalman filter with observation variance $r^2 = 7$ and $\omega = 0.02$. Display a plot showing the ground truth, the Kalman filter approximation and the observations points.

Consider now the discrete model

$$x_0 = 0, \quad x_1 = 1, \quad \text{and for } k = 1, \dots, K, \quad x_{k+1} - 2x_k + x_{k-1} = \omega^2 x_k - \lambda^2 x_k^3.$$

This is the discretisation of the anharmonic 1D oscillator

$$x'' - \Omega^2 x + \Lambda^2 x^3 = 0.$$

Let $K = 1000$ and suppose that we observe for every p timesteps

$$y_k = H u_k + \xi_k, \quad \text{where} \quad H = (1, 0), \quad \xi_k \sim \mathcal{N}(0, r^2).$$

2. Implement the extended Kalman filter for the state vectors $u_k = \binom{x_k}{x_{k-1}}$, with $p = 20$, $r^2 = 7$, $\omega = 0.03$ and $\lambda = 0.0003$. Display a plot showing the ground truth, the extended Kalman approximation and the observations points.
3. Observe that the extended Kalman filter becomes unstable when p becomes too large (try $p = 50$). Implement the ensemble Kalman filter with $m = 10$ particles and $p = 50$. Display a plot showing the ground truth, the ensemble Kalman approximation and the observations points.

[7 marks]

A The discrete TV operator

TV operator In the discrete setting, derivatives are replaced by finite difference. In 2D domain, we cosnider the two directions: horizontal and vertical. The finite differences along these two directions are

$$\text{Horizontal}: (D_h u)_{i,j} = \frac{u_{i,j+1} - u_{i,j}}{\gamma_h},$$
$$\text{Vertical}: (D_v u)_{i,j} = \frac{u_{i+1,j} - u_{i,j}}{\gamma_v}.$$

Here *forward difference* is considered, and for given 2D images we have $\gamma_h = \gamma_v = 1$.

At the boundary, we set

$$(D_h u)_{i,n} = 0 \quad \text{and} \quad (D_v u)_{m,j} = 0.$$

We define the discrete gradient operator as $Du = (D_h u, D_v u)$.

In Matlab, Du can be realized by `diff` function, and

```
dh = @(u) [diff(u,1,2), zeros(m,1)]; % horizontal difference
dv = @(u) [diff(u,1,1); zeros(1,n)]; % vertical difference
```

The adjoint operator of D is $D^\top(v_1, v_2) = D_h^\top v_1 + D_v^\top v_2$ and the adjoints $D_h^\top$ and $D_v^\top$ can also be computed by the `diff` function:

```
%% transpose of gradient operators
dht = @(u) [-u(:,1), -diff(u(:,1:n-1),1,2), u(:,n-1)]; % horizontal
    transpose
dvt = @(u) [-u(1,:); -diff(u(1:m-1,:),1,1); u(m-1,:)]; % vertical
    transpose
```

Remark 1. The above is the operator form of TV operator, they also have matrix representation via *Kronecker product* as they are linear operations. Denote $\mathrm{Id}_m, \mathrm{Id}_n$ the identity operators of size m and n, respectively. Denote d_m, d_n the 1D finite difference operator for length m and n signal with reflected boundary condition. For example, d_m is defined by

$$d_m = \begin{bmatrix} -1 & 1 & & & & & \\ & -1 & 1 & & & & \\ & & & \ddots & & & \\ & & & \ddots & & & \\ & & & & -1 & 1 & \\ & & & & & -1 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix}_{m\times m}.$$

Then we have

$$\text{Horizontal}: \texttt{dh} = d_n \otimes \mathrm{Id}_m, \quad \texttt{dht} = d_n^T \otimes \mathrm{Id}_m,$$
$$\text{Vertical}: \texttt{dv} = \mathrm{Id}_n \otimes d_m, \quad \texttt{dht} = \mathrm{Id}_n \otimes d_m^T.$$

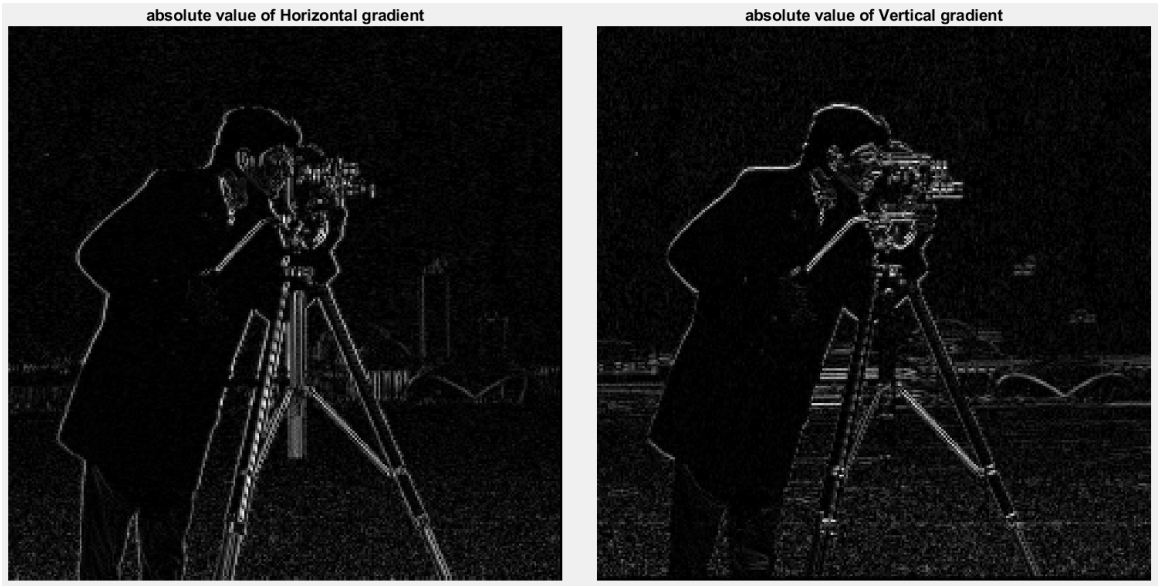
The above representation is explicit, and the matrices are sparse due to d_m, d_n. However, they are not as efficient as the operator form given above as the size of these matrices are $mn \times mn$.

Now we have Du and $D^T v$, which are

```
%% gradient of image u
gh = dh(u); % horizontal gradient
gv = dv(u); % vertical gradient
```

Note that when m, n are large enough, there holds $\|\texttt{dh}\| = \|\texttt{dv}\| = 2$ and $\|D\| = 2\sqrt{2}$.

Below we display the `absolute value` of the gradient of `cameraman` image. The reason of showing the `absolute value` is to highlight that the gradient field is sparse as most of the gradient values are (approximately) 0.



Note that `dh` does not detect horizontal edges while `dv` does not detect vertical edges.

Isotropic and anisotropic TV Denote $g^h = \texttt{dh}(u)$ and $g^v = \texttt{dv}(u)$, then

$$(Du)_{i,j} = [g^h_{i,j}, g^v_{i,j}].$$

There are also two different ways to define TV of images: anisotropic TV and isotropic TV. **In our exercises, we will use isotropic TV as this is the natural discretization of the continuous operator we have been studying, however, we mention both here for completeness.**

- **Anisotropic** The anisotropic TV is rather straightforward, simply the ℓ_1-norm of the whole gradient field, that is

$$\|Du\|_1 = \|g^h\|_1 + \|g^v\|_1 = \sum_{i,j} \left(|g^h_{i,j}| + |g^v_{i,j}|\right).$$

```
%% anisotropic TV
aTV = sum(abs(gh(:)) + abs(gv(:)));
% many functions in Matlab work both for scalar and matrix data;
% *(:)* for vecterizing a matrix into a column vector, otherwise we
    need to use two sum for sum all the elements of a matrix.
```

- **Isotropic TV** The isotropic TV considers the ℓ_2-norm of the gradient at each pixel, and then sum them together, that is

$$\|Du\|_{2,1} = \sum_{i,j} \sqrt{(g^h_{i,j})^2 + (g^v_{i,j})^2}.$$

  In Matlab, it is rather simple to compute $\|Du\|_1$,

```
%% isotropic TV
iTV = sum(sqrt(gh(:).^2 + gv(:).^2));
% ^2 for matrix means the power of the matrix, to have element-wise
    square, use .^
```

The differences of the above two definitions include geometry preference, and numerical implementation which will be discussed later. The visual illustration of these two TV are shown below.

Visually no big difference, quantitatively anisotropic has larger range of the value.