CPSC 406, Term I, 2016-17
Assignment 5, due Thursday, Nov 24

Do Questions 1 and 2, and **either** Question 3 **or** Question 4 (but **not both**). Each of the three questions is worth 40 points for a maximum total of 100.

Please show all your work: provide a hardcopy of the entire assignment (including plots); in addition, e-mail your MATLAB programs to `cs406@ugrad.cs.ubc.ca`. When e-mailing your programs, include your name and student ID in the message's title.

Do not e-mail a complete assignment: only the programs.

1. The purpose of this question is to convince you that exact line search can be far from the most efficient choice of step size for the gradient descent method.

   We are looking at the minimization of

   $$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x},$$

   where the matrix A is SPD and potentially large. The necessary and sufficient condition for a minimum is that $\mathbf{x} = \mathbf{x}^*$ satisfy $A\mathbf{x} = \mathbf{b}$. We will apply three methods:

   (i) Conjugate gradient (CG).

   (ii) Steepest descent (SD), i.e., gradient descent with the step size defined by exact line search,

   $$\alpha_k = \alpha_k^{SD} = \frac{\mathbf{r}_k^T\mathbf{r}_k}{\mathbf{r}_k^T A\mathbf{r}_k}, \quad \mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k.$$

   (iii) Gradient descent with the SD formula for step size applied only every second step, i.e., for k even use $\alpha_k = \frac{\mathbf{r}_k^T\mathbf{r}_k}{\mathbf{r}_k^T A\mathbf{r}_k}$, and for k odd use the already calculated $\alpha_k = \alpha_{k-1} = \frac{\mathbf{r}_{k-1}^T\mathbf{r}_{k-1}}{\mathbf{r}_{k-1}^T A\mathbf{r}_{k-1}}$. Call this variant HLSD.

   Furthermore, we experiment with $n \times n$ "model Poisson" matrices A defined in Example 7.1 of Ascher-Greif. Denote $N = \sqrt{n}$ and $h = 1/(N+1)$. Such a matrix has only 5 diagonals with potentially nonzero elements, and its eigenvalues are conveniently given by

   $$\lambda_{l,m} = 4 - 2(\cos(l\pi h) + \cos(m\pi h)), \quad 1 \le l, m \le N.$$

   In MATLAB a sparse representation of this matrix is obtained by the instruction

   ```
   A = delsq(numgrid('S',N+2));
   ```

(a) Derive a formula for computing the condition number $\kappa_2(A)$ without actually forming this matrix.

How does $\kappa_2(A)$ depend on n?

(b) Write a program that solves this problem using the three iterative methods specified above, where the right hand side $\mathbf{b}$ is defined to have the same value h^2 in all its n locations. Your program should use only one matrix-vector product calculation at each iteration and not ever store A or any other $n \times n$ matrix in full (i.e., the total storage requirement should be $\mathcal{O}(n)$, not $\mathcal{O}(n^2)$). Terminate the iteration for each method when the residual satisfies

$$\|\mathbf{r}_k\| \leq \texttt{tol}\|\mathbf{b}\|.$$

In addition, your program should compute the condition number and the maximum step size encountered, $\max_k \alpha_k$.

Test the program on a small example, say $N = 3$, before proceeding further(!)

(c) Apply your program for the problem instances $N = 2^l - 1,\ l = 2, 3, 4, 5, 6$, using $\texttt{tol} = 10^{-5}$. In one table, record for each l the size n, the condition number, and the number of iterations required to reduce the relative residual to below the tolerance for each of the three methods. What relationship do you observe between the condition number and the iteration count? What other observations can you make?

2. This question, as well as the next one, are concerned with recovering a function $u(t)$ on the interval $[0, 1]$ given noisy data b_i at points $t_i = ih$, $i = 0, 1, \ldots, N$, with $N = 1/h$. Because the data values are noisy, we cannot simply set $u(t_i) \equiv u_i = b_i$: knowing that $u(t)$ should be piecewise smooth, we add a regularization term to penalize excessive roughness in u. For the unknown vector $\mathbf{u} = (u_0, u_1, \ldots, u_N)^T$ we therefore solve

$$\min \phi_2(\mathbf{u}) = \frac{h}{2} \sum_{i=1}^{N} \frac{1}{2} \left[(u_i - b_i)^2 + (u_{i-1} - b_{i-1})^2 \right] + \frac{\beta h}{2} \sum_{i=1}^{N} \left(\frac{u_i - u_{i-1}}{h} \right)^2,$$

where $\beta > 0$ is a known regularization parameter.

(a) Write down the gradient and the Hessian of this objective function. To describe the regularization term use the matrix

$$W = \frac{1}{\sqrt{h}} \begin{pmatrix} -1 & 1 & & \\ & -1 & 1 & \\ & & \ddots & \ddots \\ & & & -1 \quad 1 \end{pmatrix} \in \Re^{N \times (N+1)}.$$

Show that the function $\phi_2(\mathbf{u})$ is convex.

[Guidance: write the objective function as

$$\phi_2(\mathbf{u}) = \frac{1}{2}\|D(\mathbf{u}-\mathbf{b})\|^2 + \frac{\beta}{2}\|W\mathbf{u}\|^2$$

for an appropriate *diagonal* matrix D.]

(b) Solve this problem numerically for the following problem instances. To "synthesize data" for a given N, start with

$$b_p(t) = \begin{cases} 1 & 0 \le t < .25 \\ 2 & .25 \le t < .5 \\ 2 - 100(t-.5)(.7-t) & .5 \le t < .7 \\ 4 & .7 \le t \le 1 \end{cases}.$$

Evaluate this at the grid points t_i and then add noise as follows:

```
noisev = randn(size(b_p)) * mean(abs(b_p)) * noise;
data = b_p + noisev;
```

The resulting values (and not b_p) are the data that your program "sees". An illustration is given in Figure 1.

Plot this data and the recovered curve $\mathbf{u}$ for $N = 128$ and the parameter values $(\beta, \texttt{noise}) = (10^{-3}, .01)$, $(10^{-4}, .01)$, $(10^{-3}, .1)$ and $(10^{-4}, .1)$. What are your observations?

3. Continuing Question 2, let us denote $d_i = \frac{u_i - u_{i-1}}{h}$ and $\mathbf{d} = (d_1, d_2, \ldots, d_N)^T$. The regularization term in $\phi_2(\mathbf{u})$ can therefore be written as $\frac{\beta h}{2}\|\mathbf{d}\|_2^2$.

Unfortunately, if the data function contains jump discontinuities, then these discontinuities will be smeared by the ℓ_2-regularization proposed in Question 2. So, consider instead the minimization problem

$$\min \phi_1(\mathbf{u}) = \frac{h}{2}\sum_{i=1}^{N} \frac{1}{2}\left[(u_i - b_i)^2 + (u_{i-1} - b_{i-1})^2\right] + \gamma h \|\mathbf{d}\|_1.$$

Here, $\gamma > 0$ is a regularization parameter to be specified below. The ℓ_1-regularization term in ϕ_1 is called *total variation* (TV).

To develop the method of the present Question, called *lagged diffusivity* or IRLS, we note first that the gradient of $\|\mathbf{d}\|_1$ with respect to $\mathbf{u}$ may not be bounded. So we modify $\phi_1(\mathbf{u})$ to read

$$\min_{\mathbf{u}} \phi_1^{\varepsilon}(\mathbf{u}) = \frac{h}{2}\sum_{i=1}^{N} \frac{1}{2}\left[(u_i - b_i)^2 + (u_{i-1} - b_{i-1})^2\right] + \gamma h \sum_{i=1}^{N} \sqrt{\left(\frac{u_i - u_{i-1}}{h}\right)^2 + \varepsilon},$$

where $\varepsilon = 10^{-6}$, say.

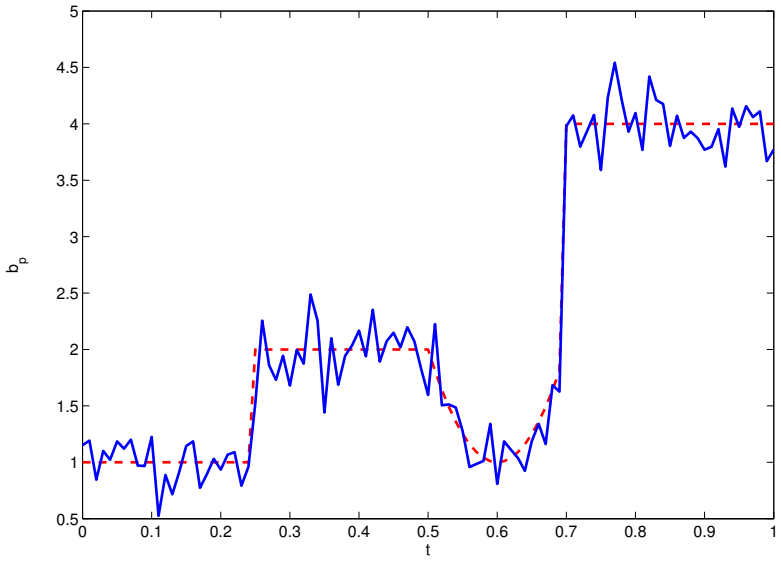

Figure 1: A depiction of the noisy function (in solid blue; the given data values are the vertices, or nodes, of this curve) and the function to be recovered (in dashed red) for Questions 2 and 3.

(a) Let $T_{1,h}$ denote what γ multiplies in the objective function ϕ_1^ε. Show that, for $0 \le j \le N$ (setting $u_{-1} = u_0$ and $u_{N+1} = u_N$),

$$\frac{\partial T_{1,h}}{\partial u_j} = \frac{u_j - u_{j-1}}{\sqrt{(u_j - u_{j-1})^2 + \varepsilon h^2}} + \frac{u_j - u_{j+1}}{\sqrt{(u_j - u_{j+1})^2 + \varepsilon h^2}}.$$

Moreover, letting

$$\hat{D} = \mathrm{diag}\{h/\sqrt{(u_j - u_{j-1})^2 + \varepsilon h^2}\}, \quad \hat{B} = \sqrt{\hat{D}},$$

it is possible to write

$$\nabla T_{1,h} = W^T \hat{B}^T \hat{B} W \mathbf{u} = W^T \hat{D} W \mathbf{u}.$$

(b) A method for solving this problem then suggests itself, where at the beginning of each iteration we fix $\hat{D} = \hat{D}(\mathbf{u})$ based on the current iterate $\mathbf{u} = \mathbf{u}_k$, and apply the usual algorithms for a linear weighted least squares functional to obtain $\mathbf{u}_{k+1}$. This is called *iteratively reweighted least squares* (IRLS).

Use this method to solve the same problems as in the previous question (i.e., same synthesized data), for $\gamma = 10^{-2}$ and $\gamma = 10^{-3}$. [In other words, with $N = 128$ try the four combinations $(\gamma, \texttt{noise}) = (10^{-2}, .01)$, $(10^{-3}, .01)$, $(10^{-2}, .1)$ and $(10^{-3}, .1)$.] Discuss your observations.

4. Instead of using the method described in Question 3 you have the option of developing a split augmented Lagrangian method (ADMM) for the same purpose.

   (a) Read the writeup `sparseII` (no points for this item, I'm afraid).

   (b) Develop an appropriate method of this sort to minimize $\phi_1(\mathbf{u})$ defined in Question 3 (without any ε).

   (c) Apply your method to solve the problem as specified in Question 3 and, where relevant, Question 2. Discuss your observations.

   [Please solve precisely one of Questions 3 or 4.]