CS5321 Numerical Optimization Homework 2

Due Nov 4

1. (20%) Let $A = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 3 & 0 \\ 1 & 0 & -2 \end{pmatrix}$
   (a) Compute the LDLT decomposition of $A = LDL^T$.
   (b) Compute the eigenvalues of A.

2. (20%) The matrix norm can be defined by vector norm as follows
$$\|A\| = \max_{\|x\|=1} \|Ax\|.$$
   For vector 2-norm, which means $\|x\| = \sqrt{x^T x}$, show that
   (a) If Q is an orthogonal matrix, $Q^T Q = I$, $\|Qx\| = \|x\|$.
   (b) If A is symmetric, $\|A\| = \max_i |\lambda_i|$, where λ_i are A's eigenvalues.

3. (60%) The Rosenbrock function $f(x, y) = (1-x)^2 + 100(y - x^2)^2$ has minimizer at $(1, 1)$. [1]
   (a) Derive the gradient and the Hessian of $f(x, y)$.
   (b) Implement the backtracking line-search method, and make it a function. Your input should at least include
      - The current solution $\vec{x}_k$.
      - The search direction $\vec{p}_k$.
      - The function that can evaluate $f(\vec{z})$ for a given $\vec{z}$.
      - The function that can evaluate $\nabla f(\vec{z})$ for a given $\vec{z}$.
   (c) Implement (1) the steepest descent method (2) Newton's method (3) CG and (4) BFGS. Use your line search algorithm to find the best step length. Use $(x_0, y_0) = (-1.2, 1.0)$ as the initial guess and compare their results. Draw the convergence figures of those four methods, whose y-axis is $\log(|\nabla f|)$ and x-axis is the number of iterations. You can use Matlab's `semilogy` for the plot.
   (Do not use any symbolic computation of Matlab, like sub or diff, or eval. Write your own function, gradient, and Hessian subroutines, and use them in your code.

[1] You can find reference of this function in Wikipedia.

The conjugate gradient algorithm

1. Given $\vec{x}_0$. Let $\vec{r}_0 = g - H_0\vec{x}_0$ and $\vec{d}_0 = \vec{r}_0$.
2. For $k = 0, 1, 2, \ldots$ until $\|\vec{r}_k\| \leq \epsilon$

$$\begin{aligned}
\alpha_k &= \frac{\vec{r}_k^T \vec{r}_k}{\vec{d}_k^T H_k \vec{d}_k} \text{ (or better line search methods to compute } \alpha_k). \\
\vec{x}_{k+1} &= \vec{x}_k + \alpha_k \vec{d}_k \\
\vec{r}_{k+1} &= \vec{r}_k - \alpha_k H_k \vec{d}_k \\
\beta_k &= \frac{\vec{r}_{k+1}^T r_{k+1}}{\vec{r}_k^T \vec{r}_k} \\
\vec{d}_{k+1} &= \vec{r}_{k+1} + \beta_k \vec{d}_k
\end{aligned}$$

3. Evaluate H_{k+1}

The SR1 algorithm

1. Given $\vec{x}_0$. Let $B_0 = I$. (The inverse Hessian approximation.)
2. For $k = 0, 1, 2, \ldots$ until $\|\nabla f_k\| \leq \epsilon$
3. Compute $\vec{d}_k = -B_k \nabla f_k$
4. Compute step length α_k.
5. $\vec{x}_{k+1} = \vec{x}_k + \alpha_k \vec{d}_k$
6. Compute $B_{k+1} = B_k + \dfrac{(\vec{s}_k - B_k\vec{y}_k)(\vec{s}_k - B_k\vec{y}_k)^T}{(\vec{s}_k - B_k\vec{y}_k)^T \vec{y}_k}$

   where $\vec{s}_k = \vec{x}_{k+1} - \vec{x}_k = \alpha_k \vec{d}_k$ and $\vec{y}_k = \nabla f_{k+1} - \nabla f_k$.

   If $1 + (\vec{s}_k - B_k\vec{y}_k)^T \vec{y}_k = 0$, let $B_{k+1} = B_k$