EN.550.661: Nonlinear Optimization I

Instructor: Daniel P. Robinson

Homework Assignment #3

Starred exercises require the use of Matlab.

Exercise 3.1**:** Define the quantity

$$\cos(\theta_k) = \frac{-g_k^T p_k}{\|g_k\|_2 \|p_k\|_2}$$

where $g_k = \nabla f(x_k)$, $B_k p_k = -g_k$, and B_k is symmetric and positive definite.

(a) Prove that if $\text{cond}(B_k) := \|B_k\|_2 \|B_k^{-1}\|_2 \leq \beta$ for some $\beta > 0$ and all $k = 1, 2, \ldots$, then

$$\cos(\theta_k) \geq 1/\beta$$

(b) Use part (a) and the result by Zoutendijk to prove that if $f : \mathbb{R}^n \to \mathbb{R}$ is $\mathcal{C}^1$ and bounded below on $\mathbb{R}^n$, $\nabla f(x)$ is Lipschitz continuous on $\mathbb{R}^n$, and the Wolfe conditions are enforced during the line search, then

$$\lim_{k \to \infty} g_k = 0.$$

Exercise 3.2**:** Show that if B is a symmetric positive-definite matrix and the vectors s and y satisfy $s^T y > 0$, then the BFGS update

$$B^+ = B + \frac{yy^T}{y^T s} - \frac{Bss^T B}{s^T Bs}$$

ensures that B^+ is also a positive-definite matrix.

Exercise 3.3***:** Write a Matlab m-function that computes a modified Newton matrix based on Algorithm 2 in the course lectures. The function call should have the form

```
[ B, flag ] = modNewton( H, beta )
```

where the input `H` is required to be a symmetric matrix and `beta` > 1 is an upper bound on the required condition number of the modified matrix. On exit, the (possibly) modified positive-definite matrix is stored in `B`, and `flag` should contain the value 0 if no modification was required and the value 1 otherwise.

Exercise 3.4*: Consider the problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\ f(x)$$

where f is a twice continuously differentiable function.

(a) Write a MATLAB m-function that minimizes a twice continuously differentiable function f using a backtracking-Armijo linesearch. The function call should have the form

```
[x,F,G,H,iter,status] = uncMIN(fun,x0,step,maxit,printlevel,tol)
```

where `fun` is of type *string* and represents the name of a Matlab m-function that computes $f(x)$, $\nabla f(x)$, and $\nabla^2 f(x)$ for some desired function f; it should be of the form

```
[F,G,H] = fun(x)
```

where for a given value x it returns the values of the function, gradient, and Hessian, respectively. The parameter `x0` is an initial guess at a minimizer of f, `step` indicates how the search direction should be computed, `maxit` is the maximum number of iterations allowed, `printlevel` determines the amount of printout required, and `tol` is the final stopping tolerance. If `step` has the value 0, then a steepest-descent search direction should be used; otherwise, a modified-Newton search direction should be computed using your m-file from Exercise 3.3. In the code, if the parameter `printlevel` has the value zero, then no printing should occur; otherwise, a single line of output is printed (in column format) per iteration. On output, the parameters `x`, `F`, `G`, and `H` should contain the final iterate, function value, gradient vector, and Hessian matrix computed by the algorithm. The parameter `iter` should contain the total number of iterations performed. Finally, `status` should have the value 0 if the final stopping tolerance was obtained and the value 1 otherwise.

(b) Write a separate MATLAB m-file with function declaration `[F,G,H] = fun(x)` that returns the value F, gradient G, and Hessian H at the point $x \in \mathbb{R}^2$ of the function

$$f(x) = 10(x_2 - x_1^2)^2 + (x_1 - 1)^2.$$

Use your m-function `uncMIN.m` from part (a) to minimize f with input `step` $= 0$ and then a second time with `step` $= 1$. In both cases, start with $x_0 = (0,0)$. Comment on your results.