最佳化方法與應用 MA5037

Homework Assignment 3

Due Jan. 17. 2024

注意：繳交作業時請同時附上程式以及相關執行結果（第二題第二小題以及第三題）。

Problem 1. 在這個問題中我們將建立一個取名為 “Wolfe_step” 的 matlabl 函數，它的功能是在給定函數 f 及它的 gradient ∇f（以及一些其它的參數）後能給出滿足 Wolfe condition 以及 Strong Wolfe condition 的步長 α。給定函數 $f : \mathbb{R}^n \to \mathbb{R}$ 以及它的 gradient g（或是逼近 ∇f 的函數）、第 k 步起點 x_k 以及第 k 步下降方向 p_k 以及滿足 $0 < c_1 < c_2 < 1$ 的常數 c_1 與 c_2、則

$$\alpha = \text{Wolfe\_step}(f, g, x_k, p_k, c_1, c_2, \text{SWITCH})\,.$$

其中 SWITCH $= 0$ 時則 α 滿足 Wolfe condition

$$\begin{aligned} f(x_k + \alpha p_k) &\leqslant f(x_k) + c_1 \alpha g(x_k)^{\mathrm{T}} p_k\,, \\ g(x_k + \alpha p_k)^{\mathrm{T}} p_k &\geqslant c_2 g(x_k)^{\mathrm{T}} p_k\,, \end{aligned}$$

而 SWITCH $= 1$ 時則 α 滿足 Strong Wolfe condition

$$\begin{aligned} f(x_k + \alpha p_k) &\leqslant f(x_k) + c_1 \alpha g(x_k)^{\mathrm{T}} p_k\,, \\ |g(x_k + \alpha p_k)^{\mathrm{T}} p_k| &\leqslant c_2 |g(x_k)^{\mathrm{T}} p_k|\,, \end{aligned}$$

試依照下列的方法寫下 Wolfe_step 這個函數：

依序確認 1, 0.5, 這兩個數是否有滿足 Wolfe condition/Strong Wolfe condition 的數，第一個滿足條件的數設為 α。若上述數字沒有任何一個滿足 Wolfe condition/Strong Wolfe condition，改依序確認 0.75, 0.25 這兩個數字是否滿足該條件，第一個滿足條件的數設為 α。若無，則改依序確認 0.875, 0.625, 0.375, 0.125 這四個數是否有滿足 Wolfe condition/Strong Wolfe condition，第一個滿足條件的數設為 α，依此方式類推進行，直到找到 α 為止。注意到上述的數字在二進位之下，是依序搜尋：

$$1, 0.1, 0.11, 0.01, 0.111, 0.101, 0.011, 0.001, \cdots\,.$$

注意到這個方法不是很有效率找到滿足 Wolfe condition/Strong Wolfe condition 步長的方法！事實上一般很難有快速的方法找到滿足 Wolfe condition/Strong Wolfe condition 的步長，因此才會有 back-tracking 的方式（加上取 c_1 很靠近 0 和 c_2 很靠近 1 來取巧）。

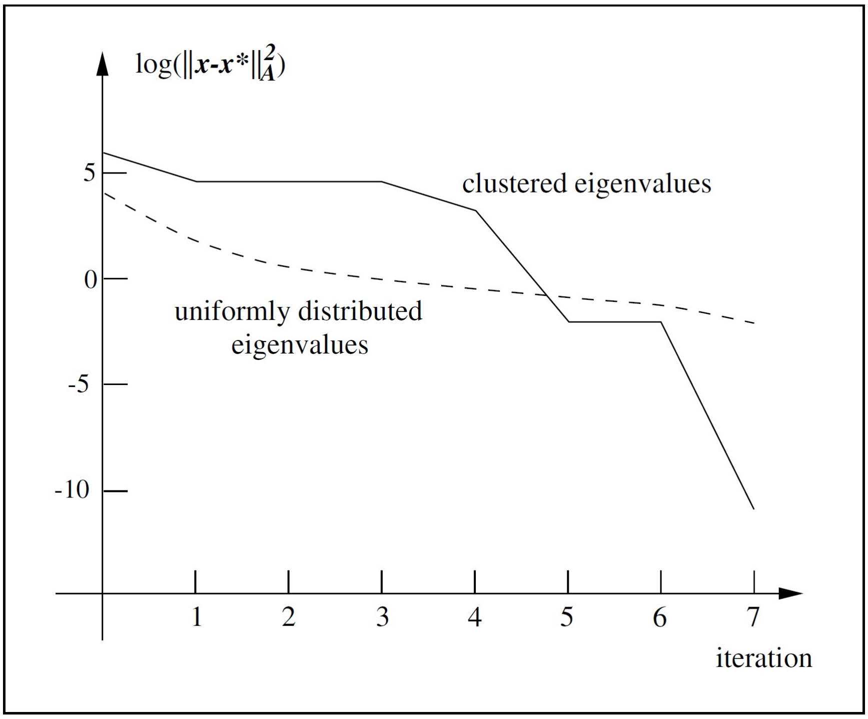
Problem 2. Let A be an $n \times n$ positive definite matrix, and b be an $n \times 1$ column vector.

1. Consider solving $Ax = b$ using the conjugate gradient method. Write a matlab® function named CG with
   (a) input variables: A, b, the initial guess x_0;
   (b) output variables: the collection of iterates x; that is, $x = [x_1, x_2, \cdots, x_n]$ in which each iterate x_k is a column vector. If the solution x_* if found at the ℓ-th step for some $\ell < n$, $x = [x_1, x_2, \cdots, x_{\ell-1}, x_*]$.

   In other words,

$$x = \text{CG}(A, b, x_0)\,.$$

2. Randomly generate a positive definite matrix A with clustered eigenvalues. Use the code in part 1 to observe $\log_{10}(\|x_k - x_*\|_A^2)$ and see if you obtain something like



Problem 3. Let $f : \mathbb{R}^2 \to \mathbb{R}$ be a function given by

$$f(x, y) = xe^{-x^2-y^2}$$

whose gradient and Hessian are given respectively by

$$(\nabla f)(x, y) = e^{-x^2-y^2}\left[1 - 2x^2; -2xy\right]$$

and

$$(\nabla^2 f)(x, y) = e^{-x^2-y^2}\begin{bmatrix} 4x^3 - 6x & 4x^2y - 2y \\ 4x^2y - 2y & 4xy^2 - 2x \end{bmatrix}.$$

1. Implement the nonlinear conjugate gradient method Algorithm FR, PR and HS with $x_0 = (0.5, 0.5)$ and stopping criteria

$$\|\nabla f_k\|_\infty < 10^{-5}(1 + |f_k|)\,.$$

   Make a table like the one given in the last slide of Chapter 5.

2. Implement the DFP and BFGS methods with H_0 given by equation (23) on page 81 in the slide of Chapter 6, and choose an $x_0 = (0.5, 0.5)$. Compare the Hessian matrices with B_k generated by the DFP and the BFGS methods. Is the DFP method less effective than the BFGS method in approximating the Hessian?