MA40050 Numerical Optimisation and Large–Scale Systems
Assessed Coursework 2019–20

Set : Friday 27th March 2020

Due : Friday 1st May 2020 (at 17.00 pm on Moodle)

This assignment is worth 25% of the total assessment for MA40050. In the questions below, the notation **[marks]** indicates the marks available for each part out of a total of 25.

The time required for the assignment is estimated to be about **10-15 hours**, on the assumption that you have studied and understood your lecture notes, and have done the problem sheets for this unit! If you have not done this then the assignment may take you longer.

Any form of cheating is a serious breach of University regulations and will result in disciplinary procedures. You should not discuss the details of your work with anyone else. The work which you hand in must be your own. You should be prepared to explain anything which you write to an examiner if asked to do so. In particular, if it is discovered that all or part of your code or your written work has been copied, both parties involved risk a severe penalty and might lose all their marks on the assignment.

You may use any results which have been proved in lectures or on problem sheets, provided you quote them clearly. Cite any sources that you use to complete this assignment.

Answer all the questions and show your working clearly. In particular explain your derivations explicitly and state clearly which results you use from the lectures and problems sheets. It is preferable to typeset your work in LaTeX. Scans/photos of handwritten solutions are also acceptable provided they are evenly lit and clear enough to read. **I will reduce your overall mark by up to five marks if the scans/photos are difficult to read**. Make sure your work is **well-presented** and **readable**, in particular, use **full sentences**. Make sure any figures are labelled correctly and have visible lines. You will lose marks for incoherent arguments and derivations. Matlab code should be well commented. **I will reduce your overall mark by up to five marks if anything is unclear, unreadable, incoherent, not commented, not labelled, not concise or not well written.**

How to hand in: Upload your coursework through the Moodle page of the course by the deadline. The submission should be a single zip file which you should name with your BUCS user name (this work is not anonymously marked). This zip file should have two folders: written and code. For the **written** part, any scans/photos should be named in an increasing order (e.g. by page number e.g 'pg01', 'pg02' or auto generated numbers). For the **code** part, implement Matlab functions when instructed to do so. The 'code' folder should include a README file with your submission that explains how to run your programs. I will only type the name of the main program to try out your code.

1. Let $A \in \mathbb{R}^{N\times N}$ be spd and let $x \in \mathbb{R}^N$, $x \neq 0$. Prove that

$$(x^T x)^2 \leq (x^T A x)\,(x^T A^{-1} x)$$

When does equality hold? **[2]**

[**Hint:** Substitute $u = A^{1/2}x$ and $v = A^{-1/2}x$ and apply the Cauchy-Schwarz inequality.]

2. Suppose $f \in C^1(\mathbb{R}^N)$ and $x_n \in \mathbb{R}^N$ with $g_n := \nabla f(x_n) \neq 0$. Let $H_n = H_n^T \in \mathbb{R}^{N\times N}$ be an approximation of $\nabla^2 f(x_n)$ and recall the quadratic model

$$m_n(x) := f(x_n) + g_n^T(x - x_n) + \tfrac{1}{2}(x - x_n)^T H_n (x - x_n),$$

as well as the (quasi-)Newton point x_n^N and the unidirectional minimiser x_n^U of m_n, defined respectively by:

$$x_n^N := x_n - H_n^{-1} g_n \quad \text{and} \quad x_n^U := x_n - \alpha_n^U g_n \quad \text{with} \quad \alpha_n^U := \frac{|g_n|^2}{g_n^T H_n g_n}.$$

(a) Prove that, if $H_n > 0$, then

$$\gamma := (x_n^N - x_n^U)\cdot(x_n - x_n^U) \leq 0.$$

[**Hint:** Use the inequality in Question 1. From now on you may use without proof that $\gamma < 0$, unless $x_n^U = x_n^N$.]

(b) Prove that, if $\gamma < 0$, then $|x_n^N - x_n| > |x_n^U - x_n|$. Deduce that then, for every $\delta \leq |x_n^N - x_n|$, there exists a unique point $x_n^D(\delta)$ on the dogleg path

$$\Gamma_n := \{x_n + t(x_n^U - x_n) : 0 \leq t \leq 1\} \cup \{x_n^U + t(x_n^N - x_n^U) : 0 \leq t \leq 1\},$$

such that $|x_n^D(\delta) - x_n| = \delta$.

[**Hint:** You may argue geometrically and you may assume without proof that the unique point is given by

$$x_n^D(\delta) := \begin{cases} x_n - \dfrac{\delta}{|g_n|} g_n, & \text{if } \delta \leq |x_n^U - x_n|, \\ x_n^U + t_\delta(x_n^N - x_n^U), & \text{if } |x_n^U - x_n| < \delta \leq |x_n^N - x_n|, \end{cases}$$

with

$$t_\delta := \frac{\gamma + \sqrt{\gamma^2 + |x_n^N - x_n^U|^2(\delta^2 - |x_n^U - x_n|^2)}}{|x_n^N - x_n^U|^2}. \qquad]$$

(c) Prove that, if $\gamma < 0$, then the function of δ given by $m_n(x_n^D(\delta))$ is strictly decreasing for $|x_n^U - x_n| \leq \delta \leq |x_n^N - x_n|$.

[**Hint:** Consider the function $\phi(t) := m_n(x_n^U + t(x_n^N - x_n^U))$ on the segment of Γ_n from x_n^U to x_n^N, and show that $\phi'(t) < 0$, for all $t \in [0,1]$. Then deduce the result.]

[5]

3. Implement in `Matlab` the dogleg method for the trust region **subproblem** (20) described in Section 5.4 of the lecture notes.

   As input, the function `dogleg.m` should take x_n, $f(x_n)$, g_n, H_n and the current trust region radius Δ_n. The output of the function should be the dogleg solution $x_n^D(\Delta_n)$ to the trust region subproblem (20), as well as the value $m_n(x_n^D(\Delta_n))$ of the model evaluated at the dogleg solution.

   Comment your code and make sure it works correctly. Describe in a couple of sentences (in a comment at the top of this file) how you tested your code.

   Implement this function in a file `dogleg.m` **[4]**

4. To approximate the Hessian in our trust region method we consider a symmetric rank-1 (SR1) quasi-Newton update of the form

$$H_{n+1} = H_n + \sigma v v^T, \quad \text{for some } \sigma \in \{-1, 1\} \text{ and } v \in \mathbb{R}^N.$$

   (a) Let $d_n := x_{n+1} - x_n$ and $y_n := \nabla f(x_{n+1}) - \nabla f(x_n)$, and suppose $d_n^T(y_n - H_n d_n) \neq 0$. Prove that if H_{n+1} satisfies the secant condition $H_{n+1} d_n = y_n$, then

$$\sigma = \operatorname{sign}[d_n^T(y_n - H_n d_n)] \quad \text{and} \quad v = \delta(y_n - H_n d_n) \quad \text{with} \quad \delta = \pm |d_n^T(y_n - H_n d_n)|^{-1/2}.$$

   Hence deduce that

$$H_{n+1} = H_n + \frac{(y_n - H_n d_n)(y_n - H_n d_n)^T}{d_n^T(y_n - H_n d_n)} \tag{1}$$

   is the only symmetric rank-1 update which satisfies the secant condition.

   (b) Prove that, if $d_n^T(y_n - H_n d_n) = 0$ then a symmetric rank-1 update that satisfies the secant condition exists if, and only if, $y_n = H_n d_n$.

   (c) Using the Sherman–Morrison–Woodbury formula (or otherwise), prove that, if H_n is invertible and $d_n^T(y_n - H_n d_n) \neq 0$, then H_{n+1} is invertible and we have

$$H_{n+1}^{-1} = H_n^{-1} + \frac{(d_n - H_n^{-1} y_n)(d_n - H_n^{-1} y_n)^T}{(d_n - H_n^{-1} y_n)^T y_n}. \tag{2}$$

   [6]

5. Question 4 motivates the following safe-guarded SR1 update procedure for H_{n+1}:

   Given: (small) $\eta > 0$
   if $|d_n| = 0$ **or** $|d_n^T(y_n - H_n d_n)| < \eta |d_n| |y_n - H_n d_n|$ **then**
   $H_{n+1} = H_n$
   else
   H_{n+1} is updated using formula (1)
   end if

   Implement a safe-guarded SR1 update algorithm in `Matlab` for computing H_{n+1}, as well as its inverse H_{n+1}^{-1}. As input, the function `sr1.m` should take H_n, H_n^{-1}, d_n, y_n and η. The output of the function should be H_{n+1} and H_{n+1}^{-1}.

Comment your code and make sure it works correctly. Describe in a couple of sentences (in a comment at the top of this file) how you tested your code.

Implement this function in a file `sr1.m`

[4]

6. Modify your implementation of the Trust Region algorithm (Algorithm 5.2) from Problem Sheet 5 (or use my model code) by using

   (i) the dogleg solution of the trust region subproblem (20) at x_n instead of the Cauchy point, i.e. $\widetilde{x}_n = x_n^D(\Delta_n)$ and

   (ii) the safe-guarded SR1 updates for H_{n+1} and H_{n+1}^{-1} instead of the exact Hessian $\nabla^2 f(x_{n+1})$ and its inverse, for $n \geq 0$.

   Compute $x_n^D(\Delta_n)$ by calling the function `dogleg.m` developed in Question 3, and update H_{n+1} and H_{n+1}^{-1} by calling the function `sr1.m` developed in Question 5.

Implement this function in a file `tr_dogleg.m`

Choose $x_0 = (-1.2, 1)^T$, $\Delta_0 = 0.2$, $H_0 = I$, $\Delta_{\max} = 1$, $\rho_{ac} = 0.125$ and $\eta = 10^{-6}$ and apply the algorithm to find the minimum of the Rosenbrock function

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

Print the trust region radius Δ_n and the actual step length $|x_{n+1} - x_n|$ used at each iteration. What do you observe? Compare the convergence to the methods we implemented and tested on the problem sheets. Is the method sensitive to the choice of the parameters $\Delta_{\max}$, ρ_{ac} and η?

This commentary should be included in your 'written' answers

Visualise the solution path using the postprocessing function `visual.m` provided on course webpage (`https://www.pranavsingh.co.uk/ma40050`) as part of the model solutions to Problem Sheet 5.

Add the plot produced by `visual.m` to visualise the solution path produced by your code as a file `tr_dogleg_rosen.fig`

[4]

Pranav Singh, March 27, 2020 (`ps2106@bath.ac.uk`).