COL726 Assignment 3

Due date: 17 July, 2023

All answers should be accompanied by a rigorous justification, unless the question explicitly states that a justification is not necessary.

In some questions, earlier parts ask you to prove results that may be useful in later parts. If you are unable to prove the earlier results, you are still allowed to use them to complete the derivation in the later parts.

1. Suppose f is a smooth function and p is its interpolating polynomial on equally spaced points $t_1, t_2, \dots, t_n$. Denote $h = t_{i+1} - t_i$ and $M = \max_{t \in [t_1, t_n]} |f^{(n)}(t)|$. Using the fact that $f(t) - p(t) = \frac{1}{n!} f^{(n)}(\theta) \cdot (t - t_1)(t - t_2) \cdots (t - t_n)$ for some $\theta \in [t_1, t_n]$, prove that

   $$|f(t) - p(t)| \leq \frac{Mh^n}{4\binom{n}{i}} \quad \text{when } t_i < t < t_{i+1}.$$

   Note: This explains why the error is usually larger near the ends of the domain, because the denominator is small when i is close to 0 or n.

2. (a) Derive the general form of a 2-point interpolatory quadrature rule for $\int_{-1}^{1} f(x)\, \mathrm{d}x$, using nodes x_1 and x_2 with $-1 \leq x_1 < x_2 \leq 1$. For what values of x_1, x_2 is the quadrature rule stable?

   (b) Is it possible to choose the nodes so that the quadrature rule has degree 3? If yes, give those values of x_1, x_2. If not, explain why.

3. In class, we have stated the second-order condition for convexity: *A twice-differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if its Hessian is positive semidefinite everywhere, i.e. $\nabla^2 f(\mathbf{x}) \succeq 0$ for all $\mathbf{x} \in \mathbb{R}^n$.* Prove both directions of this equivalence. You may use the first-order condition for convexity without proving it.

   Hint: Try using Taylor's theorem as given in class. Partial marks will be given if you prove the desired result only for the one-dimensional case $f : \mathbb{R} \to \mathbb{R}$.

4. Let $f : \mathbb{R} \to \mathbb{R}$ be a twice-differentiable convex function with a unique minimizer $x^* \in \mathbb{R}$, and bounded second derivative, $f''(x) \leq M$ for all $x \in \mathbb{R}$.

   (a) Prove that $x^{(0)}, x^{(1)}, x^{(2)}, \dots$ converges to x^* if and only if $f'(x^{(k)})$ converges to zero.

   (b) Suppose $f(x^{(0)}) > f(x^{(1)}) > f(x^{(2)}) > \cdots$. Is this sufficient to guarantee that $x^{(k)} \to x^*$ as $k \to \infty$? Justify your answer with a proof or a counterexample.

5. Consider a function $f : \mathbb{R}^n \to \mathbb{R}$ of the form (note the absolute value signs!)

$$f(\mathbf{x}) = \sum_{i=1}^{k} |\mathbf{a}_i^T \mathbf{x} + b_i|.$$

Explain why f is convex but cannot be minimized effectively with gradient descent or Newton's method. Then, prove that the smoothed function $\hat{f}(\mathbf{x}) = \sum_{i=1}^{k} \sqrt{(\mathbf{a}_i^T \mathbf{x} + b_i)^2 + \epsilon}$ is convex for any constant $\epsilon > 0$. For full marks, do so without calculating any Hessians.

6. It is sometimes desirable to perform data fitting using a norm other than the 2-norm. For example, let us consider minimizing $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_p$ for some $p > 1$, or the equivalent problem of minimizing $f(\mathbf{x}) = (\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_p)^p = \sum_{i=1}^{m} |\mathbf{a}_i^T \mathbf{x} - b_i|^p$, where $\mathbf{a}_i \in \mathbb{R}^n$ are rows of $\mathbf{A}$.

   (a) Derive an expression for the gradient $\nabla f(\mathbf{x})$, or its components.

   Implement a Python function `gd(A, b, p, x0)` to find the optimal point $\mathbf{x}^*$ using gradient descent with backtracking line search, terminating when $\dfrac{\|\nabla f(\mathbf{x}^{(k)})\|}{\|\nabla f(\mathbf{x}^{(0)})\|} < 10^{-6}$.

   (b) Implement Newton's method (as a function `newton`) to solve the same problem. Your function should use the same arguments and the same termination criterion as `gd`.

   Note: You may use Numpy/Scipy's built-in functions for solving linear systems. If the Hessian is too complicated to derive, you may estimate it via centered differences applied to the gradient, then ensure symmetry via $\mathbf{H} \leftarrow \frac{1}{2}(\mathbf{H} + \mathbf{H}^T)$.

   (c) Generate some test data by choosing a degree-5 polynomial, sampling its values at a large number of points, and adding some zero-mean noise to the values (using e.g. `numpy.random`). Solve the minimization problem for $p = 1.25,\ 2,\ 5$ using the zero polynomial as the initial guess. Plot the data, the original polynomial, and the three optimized polynomials on a single plot (similar to B&V Figure 6.5). For each value of p, make a plot of $\dfrac{\|\nabla f(\mathbf{x}^{(k)})\|}{\|\nabla f(\mathbf{x}^{(0)})\|}$ as a function of iteration number k for both your algorithms.

   You may observe that when the noise follows a uniform distribution, a large value of p better recovers the original polynomial, while if the noise is heavy-tailed (e.g. following a Laplace distribution), small p does better.

Submission: You should submit (i) a PDF of your answers to all questions, and (ii) a Python file containing the functions required in Question 6. You can send both files to me by email.