COL726 Assignment 5

18 – 31 March, 2022

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

1. As mentioned in class, interpolation in the Newton basis can be performed in a "streaming" manner, taking the sample points one at a time.

   (a) Design the algorithm to carry this out. Explain (i) what data you will store to represent the interpolating polynomial, (ii) how you will update the representation given a new point (t_n, y_n), and (iii) how you will evaluate the polynomial at an arbitrary point t. Both the update and the evaluation should take $O(n)$ time.

   (b) Implement your algorithm as a Python class `StreamingNewton` which provides a constructor `__init__(self, t0, y0)`, an update method `insert(self, tn, yn)`, and an evaluation method `evaluate(self, t)` which returns $p(t)$.

2. Let's understand why polynomial interpolation has huge oscillations near the ends of the interval. Recall that at <u>any</u> point $t \in [t_1, t_n]$, the error $|f(t) - p(t)|$ is bounded above by a multiple of $w(t) = (t - t_1)(t - t_2) \cdots (t - t_n)$. Show that for uniformly spaced points, $|w(t)|$ is much bigger near the ends than in the middle. Specifically, assuming $n = 2k$, derive a quantitative lower bound on $\dfrac{\max_{t \in [t_1, t_2]} |w(t)|}{\max_{t \in [t_k, t_{k+1}]} |w(t)|}$.

   <u>Hint:</u> For <u>one</u> part, you may need $\max_{t \in [a,b]} |(t-a)(t-b)| = (b-a)^2/4$.

3. Consider the (very easy) integral $I = \int_0^1 x^2 \, dx = \frac{1}{3}$.

   (a) Suppose we divide $[0, 1]$ into n subintervals $[x_0, x_1], [x_1, x_2], \dots, [x_{n-1}, x_n]$ with $x_i = \frac{i}{n}$, and estimate I via the Riemann sum $S = \sum_{i=1}^{n} \xi_i^2 \Delta x$ for some chosen points $\xi_i \in [x_{i-1}, x_i]$ and $\Delta x = \frac{1}{n}$. Find S in closed form for (i) $\xi_i = x_{i-1}$, (ii) $\xi_i = x_i$, and (iii) $\xi_i = \frac{1}{2}(x_{i-1} + x_i)$. More importantly, explain why the third choice gives a much better estimate of I.

   (b) Find a <u>two</u>-point quadrature rule which exactly integrates not just $f(x) = x^2$ but any quadratic $f(x) = ax^2 + bx + c$ on $[0, 1]$. Is it possible to place the quadrature points symmetrically, i.e. $x_1 + x_2 = 1$?

4. For a function $f : \mathbb{R}^2 \to \mathbb{R}$, give a finite difference scheme to compute the mixed derivative

$\dfrac{\partial^2 f}{\partial x_1 \partial x_2}$, and analyze its order of accuracy using the multivariate Taylor series

$$f(\mathbf{x}+\mathbf{h}) = f(\mathbf{x}) + \sum_i \frac{\partial f}{\partial x_i} h_i + \frac{1}{2} \sum_i \sum_j \frac{\partial^2 f}{\partial x_i \partial x_j} h_i h_j + O(\|\mathbf{h}\|^3).$$

5. (a) For any two convex sets $C_1, C_2 \subset \mathbb{R}^n$, their sum $C_1 + C_2 = \{\mathbf{x}_1 + \mathbf{x}_2 : \mathbf{x}_1 \in C_1, \mathbf{x}_2 \in C_2\}$ is also convex. Prove this fact from first principles, i.e. show that for any $\mathbf{x}, \mathbf{y} \in C_1 + C_2$ and any $\theta \in [0,1]$, we have $\theta \mathbf{x} + (1-\theta)\mathbf{y} \in C_1 + C_2$.

   (b) For a convex function $f : \mathbb{R}^n \to \mathbb{R}$, show that the function $g(\mathbf{x}) = \inf_{\mathbf{y} \in \mathbb{R}^n} \big(f(\mathbf{y}) + \frac{1}{2}\|\mathbf{x}-\mathbf{y}\|_2\big)$ is also convex.

   Hint: There is a very short proof using the result of part (a).

6. The mylar balloon problem[1] asks for the curve of unit length that, when revolved about the z-axis, generates a surface with the maximum volume. Optimization is not in the scope of this assignment, but let's see how we would even represent such an arbitrary shape numerically.

   (a) Suppose we describe the curve via the angle of the tangent θ with the x-axis, as a function of length s along the curve. Then one can show by geometrical considerations that $\theta(0) = 0, \theta''(0) = 0, \theta(1) = \pi/2, \theta''(1) = 0$. Given $n-1$ intermediate samples $\theta_i = \theta(\frac{i}{n})$ for $i = 1, \ldots, n-1$, implement a natural cubic spline class, `NaturalCubic`, which takes this list in its constructor and provides an `evaluate(s)` method that returns $\theta(s)$ at any point in $[0,1]$.

   (b) The points on the curve are then given by $x(s) = \int_0^s \cos\theta(s)\,\mathrm{d}s$, $z(s) = \int_0^s \sin\theta(s)\,\mathrm{d}s$. Implement a function `(x, z) = evaluate_point(theta, s)` to compute these integrals. Then implement `V = volume(theta)` which computes the volume of the surface of revolution, $V = \int_0^1 \pi x(s)^2 z'(s)\,\mathrm{d}s$. Here `theta` should be a `NaturalCubic` object.

   Note: You can use any quadrature rule, though I suggest the composite Simpson's rule with subintervals of length $1/n$ to match the underlying spline.

   (c) Would this approach work well to model the curve that generates a cylinder of radius $\frac{1}{2}$, i.e. $\theta(s) = 0$ if $s \le \frac{1}{2}$, $\pi/2$ otherwise? Explain why or why not.

   For the special case $\theta_i = \frac{\pi i}{2n}$, verify that your $\theta(s)$ is precisely linear, the points (x, z) lie close to a circular arc of radius $r = 2/\pi$ (by plotting them), and the computed volume converges to $\frac{2}{3}\pi r^3 = 0.540379\ldots$ as n increases. Include a plot of the curve $(x(s), z(s))$ for some moderate n, and a log-log plot of $|V - \frac{2}{3}\pi r^3|$ as a function of n.

[1] `https://en.wikipedia.org/wiki/Mylar_balloon_(geometry)`

Bonus question (no marks): Later, you could try applying an optimization algorithm on the variables $\theta_1, \ldots, \theta_{n-1}$ to maximize V. The optimal volume is around $0.609262\ldots$.

Collaboration policy: Refer to the policy on the course webpage.

If you collaborated with others to solve any question(s) of this assignment, give their names in your submission. If you found part of a solution using some online resource, give its URL.

Submission: This assignment has two submission forms on Gradescope. In one form, you have to submit a PDF of your answers for all questions, and in the other you have to submit your code for the programming questions. Both submissions must be uploaded before the assignment deadline.

Code submissions should contain two files `a5q1.py` and `a5q6.py` which contain the requested functions and any helper functions. You are permitted but not required to include the code for producing the plots. The plots themselves should be included in your PDF.