Homework 8
APPM 4600 Numerical Analysis, Fall 2025

Due date: Friday, October 24, before midnight, via Gradescope.

Instructor: Prof. Becker
Revision date: 10/18/2025

Theme: interpolation and splines

Instructions Collaboration with your fellow students is OK and in fact recommended, although direct copying is not allowed. The internet is allowed for basic tasks (e.g., looking up definitions on wikipedia) but it is not permissible to search for proofs or to *post* requests for help on forums such as `http://math.stackexchange.com/` or to look at solution manuals. Please write down the names of the students that you worked with. Please also follow our AI policy.

An arbitrary subset of these questions will be graded.

Turn in a PDF (either scanned handwritten work, or typed, or a combination of both) to **Gradescope**, using the link to Gradescope from our Canvas page. Gradescope recommends a few apps for scanning from your phone; see the Gradescope HW submission guide.

We will primarily grade your written work, and computer source code is *not* necessary (and you can use any language you want). You may include it at the end of your homework if you wish (sometimes the graders might look at it, but not always; it will be a bit easier to give partial credit if you include your code). For nicely exporting code to a PDF, see the APPM 4600 HW submission guide FAQ.

Problem 1: Error formula for interpolation If $f \in C^\infty(\mathbb{R})$ and we have distinct nodes $\{x_0, x_1, \ldots, x_n\} \subset [0,1]$ and corresponding y-values $y_i = f(x_i)$, and p_n is the (unique) polynomial interpolant of degree at most n, then we have a formula for the error:

$$|f(x) - p_n(x)| = \frac{\left|f^{(n+1)}(\xi)\right|}{(n+1)!} \times \underbrace{\left|\prod_{i=0}^{n}(x - x_i)\right|}_{q_n(x)}$$

for some $\xi \in [0,1]$ (note: in this problem, i is always an index, not $\sqrt{-1}$). For many functions (e.g., $f(x) = e^{-x}$) the derivatives $f^{(n+1)}$ are either bounded or grow slowly with n, so in those cases, $\left|f^{(n+1)}(\xi)\right|$ is not our primary concern. This problem focuses on the term labeled q_n, and bounding how large it can be. Specifically, we define

$$q_n = \left|\prod_{i=0}^{n}(x - x_i)\right| = \prod_{i=0}^{n}|x - x_i|.$$

a) Choose the nodes **uniformly** so $x_i = ih$ for $i = 0, 1, \ldots, n$ where $h = \frac{1}{n}$. Consider the point $x = \frac{1}{2n}$. Show that $q_n(\frac{1}{2n}) = O(\frac{1}{\sqrt{n}}e^{-n})$ (this means it decays quite quickly with n, a good thing; and the decay is even faster if you account for the $\frac{1}{(n+1)!}$ term). *Hint: use Stirling's approximation of the factorial. Also, we are using big-O notation, so you can use use some upper bounds to simplify your computation.*

b) Again choose the nodes uniformly. Let's derive a similar result using a different technique[1]. Define $\varphi_n(x) = \frac{1}{n+1}\sum_{i=0}^{n}\log|x - x_i|$, so $q_n(x) = e^{(n+1)\varphi_n(x)}$. With $x_i = ih$, we'll interpret $\varphi_n(x)$ as the approximation of an integral (think of this either as a Riemann sum, or

[1] Inspired by Chapter 5 of the book "Spectral Methods in Matlab" by Lloyd N Trefethen, SIAM, 2000. Trefethen was using the interval $[-1, 1]$ and we've translated that to $[0, 1]$.

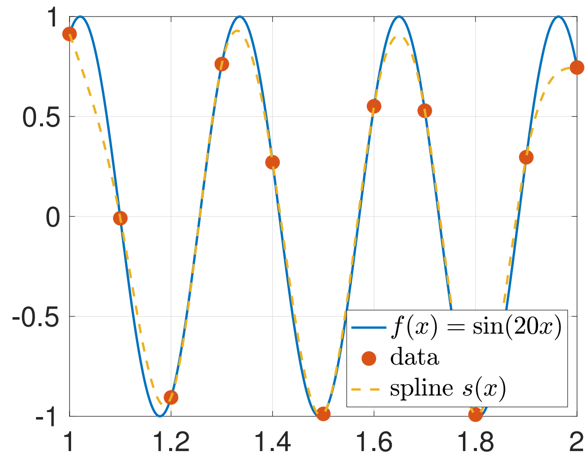

Figure 1: Example of spline for Problem 2

think of it as the expectation value if x_i are random variables sampled from the uniform distribution). Specifically, we will replace $\varphi_n(x)$ with the approximation

$$\varphi_n(x) \approx \int_0^1 \log(|x-t|)\,dt.$$

Compute this integral exactly (though to evaluate the final answer, you may have a log so you can use a computer/calculator for that) for **both** the points $x=0$ and $x=\frac{1}{2}$. Use these values to approximate q_n. For $x=0$, is this similar to your answer from part (a)? How does the error bound compare at $x=0$ with $x=\frac{1}{2}$?

c) Let's now choose the nodes **non-uniformly** using $x_i = \frac{1}{2}+\frac{1}{2}\cos(i\pi/n)$ for $i=0,1,\ldots,n$. These are the Chebyshev points (adjusted for our domain of $[0,1]$). Using the same technique as in part (b), we'll approximate

$$\varphi_n(x) = \frac{1}{n+1}\sum_{i=0}^{n} \log\left|x-\left(\frac{1}{2}+\frac{1}{2}\cos(i\pi/n)\right)\right| \approx \int_0^1 \log\left|x-\left(\frac{1}{2}+\frac{1}{2}\cos(\pi t)\right)\right|\,dt.$$

Compute this integral **numerically** (using a graphic calculator / SciPy / Matlab / Desmos / Wolfram Alpha / internet / etc.), again for **both** $x=0$ and $x=\frac{1}{2}$, and use this approximation to estimate the error bound q_n for both points. How does the error bound compare for $x=0$ with $x=\frac{1}{2}$?

Problem 2: **Splines** For this problem, we will do cubic splines. You do not need to implement these yourselves; in Matlab, you can use the curve-fitting toolbox (should be free with our campus subscription; you can also remotely use CU desktops if you need to) and the functions `csape` (to find the spline) and `fnval` (to evaluate the spline); in Python, use `scipy.interpolate.CubicSpline`.

We'll use the function

$$f(x) = \sin(20x)$$

and the interval $[1,2]$. This function will give us the values to use on the nodes. We will only consider equispaced nodes $[x_0=1, x_1=1+h, \ldots, x_n=1+nh=2]$ for $h=1/n$, and will choose different values for n.

a) Create both the **natural** cubic spline as well as the **not-a-knot** cubic spline (using the library code; you do NOT need to program the details yourself). Denote the spline by $s(x)$. We want to measure the error $|s(x)-f(x)|$ for generic points x. To estimate this error, sample 10^5 points uniformly at random from the interval $[1.01, 1.99]$ (use Matlab's `rand` or Python's `numpy.random.rand`), and report the maximum error $|s(x) - f(x)|$ for these x values. Plot this error, for both types of cubic splines, on a figure as a function of how many points n (or $n+1$) are used for nodes. Use the plot to graphically demonstrate the order of convergence of the error. *Hint*: Make sure you have a wide enough range of n values to make this convincing and make wise choices about whether axes should be linear or logarithmic.

b) Is the order of convergence you found above expected? Discuss briefly

c) Repeat problem (a) but this time sample the 10^5 test points uniformly from $[1, 2]$ (as opposed to $[1.01, 1.99]$ which we did in part (a)), and show the same kind of plot as in (a). Are the results the same or different? Discuss briefly.

d) Repeat problem (a), but get the values from the function g, where

$$g(x) = \begin{cases} f(x) & x < 1.3 \\ f(2.6 - x) & x \geq 1.3 \end{cases}.$$

Show the same kind of plot as before. Are the results the same or different? Discuss briefly.