Math 151A
HW #4, due on Friday, November 1, 2024 at 11:59pm PST.

You can use a calculator for the following problems. (If you don't have a hand calculator, Matlab or even Google should suffice.)
Please write your answers clearly. To get full credit you need to show all your work. If you are required to write code, please attach your code and all the outputs and plots to your homework.
Homework should be submitted on Gradescope.

[1] (Neville's Method) Suppose $x_j = j$ for $j = 0, 1, 2, 3$ and it is known that

$$P_{0,1}(x) = x + 1, \quad P_{1,2}(x) = 3x - 1, \quad P_{1,2,3}(1.5) = 4$$

Find

$$P_{0,1,2,3}(1.5)$$

[2]

Write down the Newton divided differences form of the interpolating polynomial for $f(x) = \sin(x)$ that passes through the points $(0, \sin(0))$, $(\pi/4, \sin(\pi/4)$, and $(\pi/2, \sin(\pi/2)$.

[3]

Determine the missing values in the divided difference table below:

$$\begin{array}{llll} x_0 = 0 & f[x_0] = -1 & & \\ x_1 = 1 & f[x_1] = ? & f[x_0, x_1] = 5 & \\ x_2 = 2 & f[x_2] = ? & f[x_1, x_2] = ? & f[x_0, x_1, x_2] = -3/2 \end{array}$$

[4] Consider the task of approximating $\sin(x)$ over the interval $[0, 1]$. If one uses a polynomial interpolant based upon $n + 1$ equispaced data points on $[0, 1]$, $x_i = ih$, $h = 1/n, i = 0, 1, \ldots, n$. How many points are required so that the error bound for the interpolant is less than 1×10^{-6}?

[5]

Suppose that $f \in C^2([x_0, x_1])$ for $x_0 < x_1$, and let $P(x)$ be the linear interpolant for f at x_0 and x_1. Using the theorem given in class on the error in polynomial interpolation, derive the following bound:

$$|f(x) - P(x)| \leq \frac{1}{8} h^2 \max_{x \in [x_0, x_1]} |f''(x)|,$$

where $h = x_1 - x_0$.

[6] Computational exercise

An important skill of a numerical analyst is to be able to utilize functions and routines previously written by others by carefully reading that code's documentation and giving these functions/routines the proper inputs. This is especially true for large scale problems and/or projects, in which writing your own code for everything from scratch would be too time-consuming. In this exercise you will practice this skill by calling three separate MATLAB routines.

In the script 'runge_phenom.m' you will find code that plots the function

$$f(x) = 1/(1 + 25x^2)$$

from $x = -1$ to $x = 1$. From class we know that using a polynomial interpolant with equispaced nodes to approximate this function can produce wild oscillations and hence a poor approximation. Your job is to use the functions 'lagrange.m', 'chebyshev_coefficients.m' and 'chebyshev_interpolant.m' (all found on CCLE), as well as the function 'spline' (built in to MATLAB) to produce interpolants for f on $[-1, 1]$.

- 'lagrange.m' produces the Lagrange polynomial for $f(x)$ at the specified interpolation nodes
- 'chebyshev_coefficients.m' and 'chebyshev_interpolant.m' will produce the so-called Chebyshev polynomial (briefly introduced in lecture 13) that interpolates $f(x)$ at the nodes $x_j = \cos(\theta_j)$, where $\theta_j = \pi j/(N-1)$ and $j = 0, \ldots, N-1$.
- 'spline' produces a piecewise-defined cubic spline for $f(x)$ at the specified interpolation nodes

To receive full credit for this problem, you must give these 4 functions the correct inputs and store the results in the vectors 'ylagrange', 'ycheb' and 'yspline', and then plot each interpolant against the original $f(x)$. The plotting code is already written for you, but you must uncomment the 'plot' commands. Save the plot to a file (a 'jpg' or 'png', so that it can be uploaded to Gradescope) for the cases $N = 7$, $N = 11$, and $N = 20$. Here N is the number of interpolation points. For each N, which interpolant gives the best results? (It may be helpful to only plot 1 or 2 interpolants at a time by commenting/uncommenting different plot commands)

Some notes:

- The scripts for Lagrange and Chebyshev interpolation must be in the same directory as 'runge_phenom.m'
- To understand how to properly utilize each function, it is helpful to type 'help [function name]' into the Command Window, for example:

  ```
  help lagrange
  ```

  will describe what the function does, i.e. what it outputs, and importantly, what inputs are needed. Since the 'spline' function is native to MATLAB, documentation can be found easily online as well. (google: 'matlab spline')