MATH 104A

Homework 3

General Instructions: You have to integrate all the problems that require coding and/or numerical computation in a single *jupyter* notebook. Make sure all your codes have a preamble which describes purpose of the code, all the input variables, the expected output, your name, and the date of the last time you modified it. Write your own code, individually. Do not copy codes! The solutions to the problems that do not require coding must be uploaded as a single pdf or as part of the *jupyter* notebook.

1. (a) Write the Lagrangian form of the interpolating polynomial P_2 corresponding to the data in the table below:

x_j	$f(x_j)$
0	1
1	1
3	-5

   (b) Use P_2 to approximate $f(2)$.

2. We proved in class that

$$\|f - p_n\|_\infty \leq (1 + \Lambda_n)\|f - p_n^*\|_\infty \tag{1}$$

   where p_n is the interpolating polynomial of f at the nodes $x_0, \ldots, x_n$, p_n^* is the best approximation of f, in the supremum (infinity) norm, by a polynomial of degree at most n, and Λ_n is the Lebesgue constant, i.e. $\Lambda_n = \|L_n\|_\infty$, where

$$L_n(x) = \sum_{j=0}^{n} |l_j(x)| \,. \tag{2}$$

   (a) Write a computer code to evaluate the Lebesgue function (2) associated to a given set of pairwise distinct nodes $x_0, \ldots, x_n$.

   (b) Consider the equidistributed points $x_j = -1 + j(2/n)$ for $j = 0, \ldots, n$. Write a computer code that uses (a) to evaluate and plot $L_n(x)$ (evaluate $L_n(x)$ at a large number of points $\bar{x}_k$ to have a good plotting resolution, e.g. $\bar{x}_k = -1 + k(2/n_e)$, $k = 0, \ldots, n_e$ with $n_e = 1000$) for $n = 4$, 10, and 20. Estimate Λ_n for these three values of n.

(c) Repeat (b) for the Chebyshev nodes $x_j = \cos\left(\frac{j\pi}{n}\right)$, $j = 0, \ldots, n$. Contrast the behavior of $L_n(x)$ and Λ_n with those corresponding to the equidistributed points in (b).

3. (a) Implement the Barycentric Formula for evaluating the interpolating polynomial for arbitrarily distributed nodes $x_0, \ldots, x_n$; you need to write a function or script that computes the barycentric weights λ_j, $j = 0, 1, \ldots, n$, first and another code to use these values in the Barycentric Formula. Make sure to test your implementation.
(b) Consider the following table of data

x_j	$f(x_j)$
0.00	0.0000
0.25	0.7070
0.52	1.0000
0.74	0.7071
1.28	-0.7074
1.50	-1.0000

Use your code in (a) to find $P_5(2)$ as an approximation of $f(2)$.

4. *The Runge Example.* Let

$$f(x) = \frac{1}{1 + 25x^2} \qquad x \in [-1, 1]. \tag{3}$$

Using your Barycentric Formula code (Prob. 3) and (4) and (5) below, evaluate and plot the interpolating polynomial p_n of f corresponding to

(a) The equidistributed nodes $x_j = -1 + j(2/n)$, $j = 0, \ldots, n$ for $n = 4$, 8, and 12.

(b) the Chebyshev nodes $x_j = \cos\left(\frac{j\pi}{n}\right)$, $j = 0, \ldots, n$ for $n = 4$, 8, 12, and 100.
As seen in class, for equidistributed nodes one can use the barycentric weights

$$\lambda_j = (-1)^j \binom{n}{j} \qquad j = 0, \ldots, n, \tag{4}$$

and for the Chebyshev nodes we can use

$$\lambda_j = \begin{cases} \frac{1}{2}(-1)^j & \text{for } j = 0 \text{ or } j = n, \\ (-1)^j & j = 1, \ldots, n-1. \end{cases} \tag{5}$$

Make sure to employ (4) and (5) in your Barycentric Formula code for this problem. To plot the corresponding p_n evaluate this at sufficiently large number of points n_e as in Prob. 2. *Note that your Barycentric Formula cannot be used to evaluate p_n when x coincides with an interpolating node*! Plot also f for comparison.

(c) Plot the error $e_n = f - p_n$ for (a) and (b) and comment on the results.

(d) Repeat (a) for $f(x) = e^{-x^2}$ for $x \in [-1, 1]$ and comment on the result.

5. (a) Equating the leading coefficient of the Lagrange form of the interpolation polynomial $p_n(x)$ with that of the Newton's form deduce that

$$f[x_0, x_1, \ldots, x_n] = \sum_{j=0}^{n} \frac{f(x_j)}{\prod\limits_{\substack{k=0 \\ k \neq j}}^{n} (x_j - x_k)}. \tag{6}$$

(b) Use (6) to conclude that divided differences are symmetric functions of their arguments, i.e. any permutation of $x_0, x_1, \ldots, x_n$ leaves the corresponding divided difference unchanged.

6. *Inverse Interpolation*. Suppose that we want to solve the equation $f(x) = 0$, for some function f which has an inverse f^{-1}. If we have two approximations x_0 and x_1 of a zero $\bar{x}$ of f then we can use interpolation to find a better approximation, $\bar{x} \approx f^{-1}(0)$, as follows. Let $y_0 = f(x_0)$ and $y_1 = f(x_1)$.

$y_j = f(x_j)$	x_j	
y_0	x_0	
y_1	x_1	$f^{-1}[y_0, y_1]$

and $p_1(0) = x_0 + f^{-1}[y_0, y_1](0 - y_0) = x_0 - y_0 f^{-1}[y_0, y_1]$. We could now define $x_2 = p_1(0)$, evaluate f at this point to get $y_2 = f(x_2)$, and then add one more row to our table to get $f^{-1}[y_0, y_1, y_2]$. Once this is computed we can evaluate $p_2(0)$ to get an improved approximation $\bar{x}$, etc. Let $f(x) = x - e^{-x}$ using the values $f(0.5) = -0.106530659712633$ and $f(0.6) = 0.051188363905973$ find an approximate value for the zero $\bar{x}$ of f by evaluating $p_1(0)$.