Numerical Analysis Homework #2

due 2021 NOV 02, 9:50 a.m.

Caution:

- To get full credit, *you must write down sufficient intermediate steps*, only giving the final answer earns you no credit!
- Please make sure that your handwriting is recognizable, otherwise you only get partial credit for the recognizable part.

1 Assignments

Answer all theoretical questions in Section 3.8.1 in the notes. Each problem weighs 5 points and the total point is thus 50.

2 `C++` programming

A. Write a `C++` package to solve the Hermite interpolation problem by designing three classes `struct InterpConditions`, `class Polynomial`, and `class NewtonInterp`, and by implementing Newton's formula and the Neville-Aitken algorithm. The interface of your design must have a natural correspondence to the underlying mathematics and must be self-explanatory. Comments must be provided to avoid ambiguities.

   (a) `InterpConditions` stores the interpolation sites and corresponding function values and function derivatives. You should enforce certain *representation invariant*s to ensure that the problem described by any object is a Hermite interpolation problem. Your data structures should be efficient.

   (b) `Polynomial` should at least have the basic methods of addition, subtraction, multiplication, evaluation at some x, derivation, output to an `ostream`, and returning the degree and coefficients of a polynomial. Feel free to implement other methods that naturally belong to polynomials. Syntactically, the class `Polynomial` must have at least two `operator`s and two `friend operator`s.

   (c) `NewtonInterp` must contain two data members, `interPoly_` and `tableOfDividedDiffs_`, to store the interpolating polynomial and the corresponding table of divided difference. `NewtonInterp` must have two ways of interpolation: (i) interpolating from scratch to overwrite the current data members and (ii) incrementally build upon the current polynomial obtained from previous interpolation conditions. This class should also have a static method that implements the Neville-Aitken algorithm.

   (d) You must test every public method of each class by clearly defining the expected outcome and by reporting your test result in the output of the test program. The usage of a test framework such as `CppUnit` is strongly encouraged but not required.

B. Test your implementation of the Newton formula on the function
$$f(x) = \frac{1}{1+x^2}$$
within $[-5,5]$ using $x_i = -5 + 10\frac{i}{n}$, $i = 0,1,\ldots,n$, and $n = 2,4,6,8$. Plot the polynomials against the exact function to reproduce the plot in the notes that illustrate the Runge phenomenon.

C. Reuse your implementation of Newton interpolation to perform Chebyshev interpolation for the function
$$f(x) = \frac{1}{1+25x^2}$$
within $[-1,1]$ on the zeros of Chebyshev polynomials T_n with $n = 5,10,15,20$. Clearly the Runge function $f(x)$ is a scaled version of the function in B. Plot the interpolation polynomials against the exact function to observe that the Chebyshev interpolation is free of the wide oscillations in the previous assignment.

D. A car traveling along a straight road is clocked at a number of points. The data from the observations are given in the following table, where the time is in seconds, the distance is in feet, and the speed is in feet per second.

Time	0	3	5	8	13
Distance	0	225	383	623	993
Speed	75	77	80	74	72

   (a) Use a Hermite polynomial to predict the position of the car and its speed for $t = 10$s.

   (b) Use the derivative of the Hermite polynomial to determine whether the car ever exceeds the 55 mi/h (81 feet per second) speed limit.

E. It is suspected that the high amounts of tannin in mature oak leaves inhibit the growth of the winter moth larvae that extensively damage these trees in certain years. The following table lists the average weight of two samples of larvae at times in the first 28 days after birth. The first sample was reared on young oak leaves, whereas the second sample was reared on mature leaves from the same tree.

Day	0	6	10	13	17	20	28
Sp1	6.67	17.3	42.7	37.3	30.1	29.3	28.7
Sp2	6.67	16.1	18.9	15.0	10.6	9.44	8.89

(a) Use Newton's formula to approximate the average weight curve for each sample.

(b) Predict whether the two samples of larvae will die after another 15 days.

The total points of `C++` programming is 50.

3 Extra credits

Additional 10% credits will be given to you if you typeset your solutions in LaTeX. You are welcome to use the LaTeX template available on my webpage. You can also get partial extra credit for typesetting solutions of *some* problems.

Note: If you choose to typeset your solutions in LaTeX, you still need to turn in a hard copy in class. In addition, please upload your latex source (.tex) and supporting files in a single zip file (**format**: `YourName_Homework2.zip`) to the course email `NumApproximation@163.com`.

4 How to submit

Your submission must contain

(a) a programming assignment report which contains all results of your program.

(b) a Makefile so that, after unzipping the tar ball, my typing of `make` under the root directory will generate an executable, which, upon invoking, generates all the results as your answers to the questions. For more explanations, you should write a pdf file that can be generated by `make doc` from latex source files.

You should send your documentation (if there is any) and your C++ source code in a single gzipped tar ball (**format:**`YourName_Homework2.tar.gz`) to the TA's email. A number of tips are given as follows.

(i) You can use either `GNU make` or `cmake` or a mixture of them.

(ii) You may use either `GNU plot` or `matlab` to plot your results.