Numerical Analysis Homework #3

due 2021 NOV 30, 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 Theoretical problems

Answer all questions in Section 4.6.1 in the notes. These eight problems weigh 4, 9, 4, 8, 10, 5, 8, and 12 points, respectively; thus the total point of this section is 60.

2 C++ programming

A. (35 points) Write a C++ package to implement three piecewise-polynomial splines (not-a-knot, complete, and periodic) and two cardinal B-splines (quadratic and cubic) by designing two template classes `Polynomial` and `Spline` and by implementing algorithms in the notes. The interface of your design should correspond naturally to the underlying mathematics and be self-explanatory. Comments must be provided to avoid ambiguities.

   (a) `Polynomial` must be templated on the order and the type of coefficients of the polynomial as follows.
   ```
   template<int Order, class CoefType>
   class polynomial;
   ```
   Your code must support `CoefType=Vec<double,2>` where `template<class T, int Dim> class Vec` has been given in the companion file `Vec.h`. On top of this, `Polynomial` should at least have the basic methods of addition, subtraction, multiplication, evaluation at some x, derivation, output to an `ostream`, and returning the 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.

   (b) `Spline` must be templated on the dimension, the order, and the type of the spline:
   ```
   template<int Dim, int Order, SplineType t>
   class Spline;
   ```
   The enumeration type `SplineType` could have values such as `ppForm` and `cardinalB` to indicate different types. For piecewise-polynomial splines, different boundary conditions should be passed as a parameter to the method
   ```
   template<int Ord> friend Spline<2,Ord,ppForm>
   fitCurve(const std::Vector<Vec<double,2>>&,
            BCType);
   ```
   where `Ord=2` and `Ord=4` must be supported and `BCType` is an enumeration type with values such as `complete`, `notAknot`, and `periodic`. You should design another friend method that looks like
   ```
   friend Spline<1,Order,t>
   interpolate(const InterpConditions&, BCType)
   ```
   where `InterpConditions` is the class you implemented in the previous homework.

   (c) You must use an implementation of `lapack` to solve the linear system.

B. (5 points) Test your implementation of complete cubic spline interpolation and not-a-knot cubic spline interpolation on the function

$$f(x) = \frac{1}{1+25x^2}$$

on evenly spaced nodes within the interval $[-1, 1]$ with $N = 6, 11, 21, 41, 81$. Compute for each N the max-norm of the interpolation error vector at mid-points of the subintervals and report the errors and convergence rates with respect to the number of subintervals.

Plot the interpolating splines against the exact function to observe that spline interpolation is free of the wide oscillations in the Runge phenomenon.

C. (5 points) Test your implementation of cardinal B-spline interpolation on the function

$$f(x) = \frac{1}{1+x^2}, \qquad x \in [-5, 5]$$

using $t_i = -6 + i,\ i = 1, \ldots, 11$ for Corollary 4.58 and $t_i = i - \frac{11}{2},\ i = 1, \ldots, 10$ for Corollary 4.59, respectively. Plot the polynomials against the exact function.

D. (5 points) Define $E_S(x) = |S(x) - f(x)|$ as the interpolation error. For the two cardinal B-spline interpolants, output values of $E_S(x)$ at the sites

$$x = -3.5, -3, -0.5, 0, 0.5, 3, 3.5$$

Output these values by a program. Why are some of the errors close to machine precision? Which of the two B-splines is more accurate?

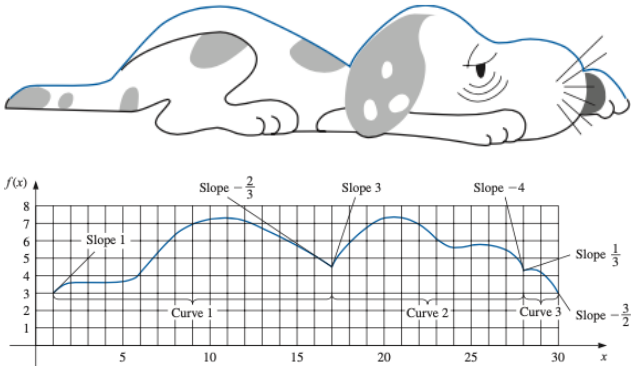
E. (5 points) The roots of the following equation constitute a closed planar curve in the shape of a heart:

$$x^2 + \left(\frac{3}{2}y - \sqrt{|x|}\right)^2 = 3.$$

Write a program to plot the heart, the program must invoke the above `fitCurve` method. The parameter

of the curve should be the *cumulative chordal length* defined in the notes. Choose $n = 10, 40, 160$ and produce three plots of the heart function. (*Hints:* Your knots should include the characteristic points and you should think about (i) how many pieces of splines to use? (ii) what boundary conditions are appropriate?)

F. (5 points) The upper portion of this noble beast is to be approximated using complete cubic splines interpolants. The curve is drawn on a grid from which the table is constructed. Use your implementation to construct the three complete cubic splines.



Curve 1				Curve 2				Curve 3			
i	x_i	$f(x_i)$	$f'(x_i)$	i	x_i	$f(x_i)$	$f'(x_i)$	i	x_i	$f(x_i)$	$f'(x_i)$
0	1	3.0	1.0	0	17	4.5	3.0	0	27.7	4.1	0.33
1	2	3.7		1	20	7.0		1	28	4.3	
2	5	3.9		2	23	6.1		2	29	4.1	
3	6	4.2		3	24	5.6		3	30	3.0	−1.5
4	7	5.7		4	25	5.8					
5	8	6.6		5	27	5.2					
6	10	7.1		6	27.7	4.1	−4.0				
7	13	6.7									
8	17	4.5	−0.67								

G. (*) Write a program to illustrate (4.37) by plotting the truncated power functions for $n = 1, 2$ and build a table of divided difference where the entries are figures instead of numbers. The pictures you generated for $n = 1$ should be same as those in Example 4.32.

The total points of `C++` programming is 60. G is an extra-credit problem that weighs 8 points.

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.