CS 370 Winter 2018: Assignment 1

Due January 25, noon

Instructor: G. Labahn Office: DC3629 e-mail: *glabahn@uwaterloo.ca*
Lectures: MWF 8:30, 11:30 Office Hours: Tues 11:00-12:00

Instructor: Y. Li Office: DC3623 e-mail: *yuying@uwaterloo.ca*
Lectures: MWF 1:30 Office Hours: Thurs 2:00-3:00

Web Site: cs370 piazza

Your assignment should be handed in electronically on UW Learn. The submission should include one pdf file containing the assignment answers and the cover sheet and all the m-files required to run the code, preferably with no folder structure (not zipped).

1. **(10 marks)** The roots of a quadratic equation, $ax^2 + bx + c = 0$, are given by

$$x_1 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}, \qquad x_2 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

when $a \neq 0$.

(a) Consider the 5-digit arithmetic of the floating point number system $F(10, 5, -10, 10)$ with truncation. Using the above formulae, carry out by hand (with the aid of Matlab or a calculator) the computation of the roots of the following quadratic equation

$$1.03x^2 + 22.41x + 0.1123 = 0. \tag{1}$$

The roots of the given quadratic, correct to five significant digits, are -0.0050123 and -21.752. What is the relative error of each of your computed roots, x_1 and x_2?

(b) A *cancellation problem* arises when applying the above formulae for any quadratic equation having the property that

$$|b| \approx \sqrt{b^2 - 4ac}.$$

When this property holds, if $b > 0$ then the quadratic formula for x_2 will exhibit cancellation. Show that an alternate formula for the roots is given by

$$x_1 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}, \qquad x_2 = \frac{c}{ax_1}.$$

(c) Redo the calculation of the roots of equation (1) by applying the above formula. Compute the relative error of the computed roots and compare against your results from part (a).

2. **(15 marks)**

Consider the sequence $\{p_n\}$ defined by $p_0 = 1,\ p_1 \ = \ \frac{4}{7}$, with p_n for $n \geq 2$ determined by

$$p_n = -\frac{41}{14} p_{n-1} + 2 p_{n-2}, \quad n \geq 2. \tag{2}$$

One can show that if

$$p_0 = 1, \quad p_1 \ = \ \frac{4}{7}$$

then

$$p_n = \left(\frac{4}{7}\right)^n$$

is an exact solution of (2) for all $n \geq 0$.

(a) Write a Matlab function to evaluate (2) for $n = 2, \ldots, 40$ using the floating point approximation $\frac{4}{7} = 0.5714285$. What happens?

(b) Carry out a stability analysis of this recursion, similar to what we did in class. Can you explain what you see? Hint: given a recursion

$$q_n + a q_{n-1} + b q_{n-2} = 0$$

with $a,\ b$ constant, then the solution to this recursion is given by

$$q_n = c_1 {x_1}^n + c_2 {x_2}^n$$

where c_1, c_2 are constants, and $x_1,\ x_2$ are roots of the equation $x^2 + ax + b = 0$.

(c) Assume that machine epsilon is $\approx 2 \times 10^{-8}$. Based on your analysis in (b), predict how many steps it will take before you will see an $O(1)$ error[1] in the floating point implementation of equation (2). (This does not have to be a rigorous argument.) How does your prediction agree with the experiment in part (a)?

Hint: assume that $p_0^{approx} = p_0^{exact}$ and that

$$\begin{aligned} (p_1)^{approx} &= (p_1)^{exact}(1+\epsilon) \\ &= (p_1)^{exact} + e_1, \qquad \text{with} \qquad e_1 = (p_1)^{exact}\epsilon \end{aligned}$$

where ϵ is machine epsilon, and e_1 is the absolute error at step 1. Assume that no other new floating point errors are introduced after step 1 (this is obviously an approximation).

3. **(10 marks)** Suppose we have a natural spline:

$$S(x) = \begin{cases} 28 + a_1 x + 9x^2 + x^3 & x \in [-3, -1] \\ a_2 + 19x + a_3 x^2 - x^3 & x \in [-1, 0] \\ 26 + 19x + 3x^2 + a_4 x^3 & x \in [0, 3] \end{cases}$$

(a) Determine the values of $a_1,\ a_2,\ \cdots,\ a_4$.

(b) Using the Lagrange basis, construct a cubic interpolation polynomial which interpolates the values of $S(x)$ at $-3, -1, 0, 3$.

[1]By $O(1)$ we mean that the error is about one.

4. **(20 marks)**

A cubic spline $S(x)$ through the points $(-1,2)$, $(1,1)$, $(2,4)$, $(5,3)$ and $(7,-1)$ and having natural boundary conditions defines a piecewise polynomial defined by

$$\begin{aligned}
S_1(x) &= a_1 + b_1(x+1) + c_1(x+1)^2 + d_1(x+1)^3 & -1 \le x \le 1,\\
S_2(x) &= a_2 + b_2(x-1) + c_2(x-1)^2 + d_2(x-1)^3 & 1 \le x \le 2,\\
S_3(x) &= a_3 + b_3(x-2) + c_3(x-2)^2 + d_3(x-2)^3 & 2 \le x \le 5,\\
S_4(x) &= a_4 + b_4(x-5) + c_4(x-5)^2 + d_4(x-5)^3 & 5 \le x \le 7.
\end{aligned}$$

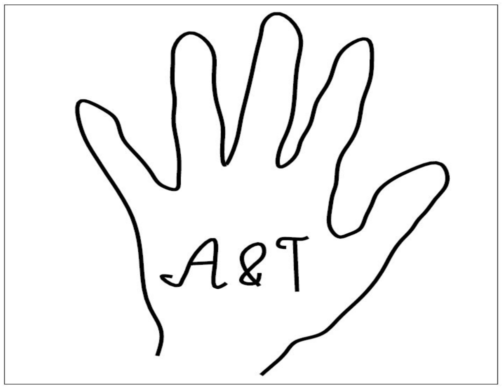
(a) Write down the 5×5 linear system of equations needed to determine the derivative values s_1, s_2, s_3, s_4, s_5 for the above cubic spline $S(x)$.

(b) Solve the linear system from part (a) and use this to determine the 16 coefficients $a_1, \ldots, d_4$ needed for the cubic spline $S(x)$.

(c) Using Matlab or Maple graph the spline $S(x)$.

5. **(20 marks)** Create parametric curves representing the outline of an image of your hand with your first and last initial separated by an ampersand inside the hand.

For example,



Follow the steps below:

(a) Start by downloading the script file **init.m** from the course website, which can be modified and used to initialize data arrays for your drawing. Use Matlab's **help** command to learn any commands that are unfamiliar to you. Given an image, you can load it and trace the image of the object directly using mouse clicks. To add your initials, write them on a piece of paper, and then hold it up to the computer screen to trace it similarly. Create your data using the mouse to select a few dozen points outlining the object and your initials. Hitting enter once begins a new sequence of points; hitting enter twice terminates the input. Points to note: You can modify the script as needed. The script calls **ginput** for each segment (i.e., each pen stroke), and uses *cell arrays* to store the data. You may also find the **save** and **load** commands useful.

(b) Using the resulting data arrays produced above, generate parametric curve representations based on smooth parametric curve interpolation as described in the course notes (see sections 3.1 and 3.2). At least one region of the sequence should be curved enough to be interesting (if not, you may change your initials). Show the output using piecewise linear and cubic splines. Specifically, you will...

Write a Matlab script splineplots.m to load the data and generate the following four (separate) figures. Use the same axis scaling v from init.m for your plots (i.e., axis(v)).

(**part1.m**) Create a plot of the interpolation data points corresponding to the crude initial shape, plotted with the '$*$' symbol and no connecting lines. The plot should have a title, and display both the axes and gridlines.

(**part2.m**) Create a plot of your initials and the object created by joining the original data points with straight lines. (The curves are not expected to look very smooth.) Include a title, and both axis and gridlines.

(**part3.m**) Create *two* smooth plots of your initials and the object using spline interpolation. Use the (approximate) arc length distance for the parameter t (described by equation 3.2 in the notes), and refine the parameter partition by a factor of 6. For the first plot use natural (or variational) end conditions, and for the second plot use not-a-knot end conditions (see functions **csape** and **fnval**). Include titles, but no gridlines or axes.

Include printouts of the generated plots as part of your written submission.