MATH 3340 - Scientific Computing Homework 5

Due: Monday, 10/26/2020, 11:59 PM

The deadline will be strictly enforced. If you do not submit in time there will be a 20% penalty for each day you're late. Remember that you are allowed to work in teams of two on this assignment. You are encouraged to prepare your work in LaTeX; a template will be provided to help you put it all together. If you choose to submit a hard copy, you may submit only one copy for a team, indicating the names of both contributors. Online submission is encouraged, however, in that case both members of a team should submit the PDF file containing their work and showing both their names.
All plots generated in this homework should have a title, legend, and labeled x and y-axes.

Instruction

1. Go to `https://www.overleaf.com` and sign in (required).
2. Open template, click *Menu* (up left corner), then *Copy Project*.
3. Go to `LaTeX/meta.tex` (the file `meta.tex` under the folder `LaTeX`) to change the section and your name, e.g.,
   - change title to `\title{MATH 3340-01 Scientific Computing Homework 5}`
   - change author to `\author{Albert Einstein \& Carl F. Gauss}`
4. For Problem 2 and 3, you can encouraged to type the solution in LaTeX. But if you want to write it on the printout, please make sure your scanned work is clear enough (failure to do so will lead to points deduction).
5. For Problem 1, you need to write function/script files, store results to output files, and save graphs to figure files. Here are suggested names for function files, script files, output files, and figure files:

Problem	Function File	Script File	Output File	Figure File
1(a)	`lagrange.m`	`hw5_p1.m`		
1(b)				`hw5_p1_b.pdf`
1(c)				`hw5_p1_c.pdf`

   Once finished, you need to upload these files to the folder `src` on Overleaf. If you have different filenames, please update the filenames in `\lstinputlisting{../src/your_script_name.m}` accordingly. You can code in the provided files in hw5.zip, and use the MATLAB script `save_results.m` to generate the output files and store the graphs to `.pdf` files automatically (the script filenames should be exactly same as listed above).
6. Recompile, download and upload the generated PDF to WyoCourses.
7. You may find LaTeX.Mathematical.Symbols.pdf and the second part of Lab 01 Slides and Lab 02 Slides helpful.

Problem 1. The purpose of this problem is two-fold. First we will construct an interpolating polynomial and compare this polynomial to the original function from which the data is extracted. Then we will investigate how the choice of nodes can affect the accuracy of a higher order interpolating polynomial.
You will notice that the interpolant will be different depending on the number and type of nodes used. Contrary to our intuition, the use of higher order polynomials does not necessarily guarantee an increase in global accuracy unless careful judgement is exercised. This should become clear when examining the plots you will generate in this problem.
We will consider the function

$$f(x) = \frac{1}{1+16x^2}$$

and construct the interpolating polynomial using the Lagrange interpolant formula given by

$$p(x) = \sum_{k=0}^{n} y_k L_k(x), \quad \text{where } L_k(x) = \prod_{\substack{j=0 \\ j\neq k}}^{n} \frac{x - x_j}{x_k - x_j}. \tag{1.1}$$

Please note that you are not supposed to use the standard form of the polynomial. Use the Lagrange form instead, as indicated above.

(a) As usual, you will create a minimum of two code files for this problem: a function file, and a script file.

(i) Write a function file, to be used in both part (b) and (c) below, that calculates the interpolating polynomial using the Lagrange interpolant in (1.1). Your function file should have the inputs:

```
xdata = the set of nodes x_k,
ydata = the values y_k = f(x_k),
x = the point, or set of points (fine grid), used to evaluate p(x)
```

and the output:

```
p = the value(s) of p at the point (or set of points) x
```

(ii) Create a script file which will execute the steps outlined in parts (b) and (c). The function and grid will be the same for both problems so at the top of this script file you should define the function $f(x)$, a set x of 100 equispaced points in $[-1, 1]$, and the associated exact values $y = f(x)$. This set will be used to evaluate and plot your interpolating polynomial. In other words, these are plotting points, not the data points.

(b) Compute the interpolating polynomial for $f(x)$ using data extracted for equispaced nodes.

(i) First you will need to generate the values needed to construct the interpolating polynomial. This is the set $\{x_k, y_k\}$, where x_k is a set of equispaced nodes in $[-1, 1]$ with

$$-1 = x_0 < x_1 < \cdots < x_n = 1.$$

and y_k is the value of the function at each node (i.e., $y_k = f(x_k)$). Use $n = 9$ (i.e., 10 nodes) and label this set `{ xdata, ydata }`.

(ii) Use the function and grid you define in part (a) to compute the values of the polynomial interpolant `p` with the information provided by `xdata` and `ydata`. Then create one plot which includes the following

- The values of the interpolant vs `x`.

- The original function $f(x)$ vs `x`.
- The set $\{x_k, y_k\}$. Plot this data set with a visible symbol (i.e., by using a marker like `*`) but don't joint these point by a line.

(c) Repeat the steps for part (b) but this time use nodes that are not equispaced in x. Instead, use the Chebyshev nodes $\{x_k\}$ given by

$$x_k = -\cos(k\pi/n), k = 0, 1, \ldots, n$$

again with $n = 9$. This will be your new data set. (In this notation the points are numbered starting from zero; to create a MATLAB vector you will need to shift the index by one.)

You may find it easier to write portions of these calculations as separate function or script files. You are more than welcome to do so.

Problem 2. This problem must be done by hand. Consider the data set and use it to approximate

k	0	1	2	3
x_k	0.1	0.3	0.5	0.7
y_k	1.01	1.35	2.11	2.95

the value the function $y = f(x)$ at the point $x = 0.45$. To do this, construct the table of values $Q_{ij}(x)$ for the recursive pointwise evaluation of the interpretation polynomial (see the typed notes section 6.2.3, page 87 and following). Display all entries in your table with 3 digit accuracy.

Problem 3. This problem must also be done by hand. You again need to approximate the value of the same function $y = f(x)$, this time at the point $x = 0.35$. Using the same data set as in Problem 2 above, compute the complete table of Newton's divided differences and then construct the resulting interpolation polynomial. Use this polynomial to approximate $y = f(x)$ at the point $x = 0.35$.