COSC2500/7500
Numerical Methods in Computational Science
Programming Exercise
Weight: 10%, Due: 3rd October, 2017.

Marcus Gallagher

Submission

For this assignment, submit (via blackboard):

- A document (pdf) giving answers to the questions attempted, including any working or comments you wish to make.
- C code files (plain text files). A single .c file (modified verison of the program given) should be sufficient for all of the Required tasks. For the additional tasks you might need to submit a different version of your program and/or multiple files.

Introduction

Spline models are a widely used technique for data interpolation (aka function approximation or regression). They are mentioned in the course notes (1.2.2) and are covered in Chap.3 of the textbook.

Preparation: Do a small amount of background reading on splines. You can also find various types of spline models implemented in Matlab. You could have a look at the documentation for "Cubic Spline Interpolation" and the curve fitting "App" if you want to see some examples. For this assignment, you don't need to have a complete understanding of all the details of implementing splines. However, it is important that you understand the key points about what a spline model is, what "input" is needed to build a spline model and what the "output" is.

On the course blackboard site, you will find a C program (`spline.c`) that performs natural cubic spline interpolation. For this assignment, you are required to use this program and make some modifications to it.

Preparation: Download the C program and check that you can compile and run it. Open the program in a text editor and study it to try and understand what it does.

Required Tasks

A grade of 1 - 5 will be awarded for completing the required tasks.

1. What is the maximum number of data points that the program can handle (i.e. the maximum value for input n) as it is written?
2. Add comments to the following lines of code, explaining what each one does:
   - `INPUT(&OK, X, A, &N);`
   - `if ((FLAG >= 1) && (FLAG <= 4)) *OK = true;`
   - `for (I=0; I<=*N; I++) {`
3. Build a natural spline model to approximate $f(x) = e^x$ using the data points $(0, 1), (1, e), (2, e^2), (3, e^3)$. Use input method 1 in the program. Cut and paste the output of the program into your document.
4. Repeat question 3 using input method 3 in the program. You will need to modify one line of code. Add this comment to the line you changed in the program: `// changed for Q2`
5. There is (at least!) one line of this program that is currently not needed. Identify this line. (Hint: the next task may give you a clue about this!)
6. The program does not actually calcuate the values of the cubic spline model at the x values given. Modify the program so that these values are calculated and printed out.
7. Build a natural spline model using the data from Exercise R1.1 from Module 1 for the course. Produce a plot of the data and the spline model. I recommend: using input method 2 and output method 2 in the C program; using Matlab to do the plot.

Additional Tasks

Attempts at these tasks can earn additional marks, but will not count towards the grade of 1 - 5 for this Programming Exercise. Completing all of these tasks does not mean that 4 marks will be obtained - the marks depend on the quality of the answers. It is possible to earn all 4 marks without completing all of these additional tasks.

1. Add a new function to the C program. The function should contain the calculation of the spline model values from Required task 6 above. It should also calculate the *error* of the model, i.e. the difference between the model and true $f(x)$ values for the data points. Finally, you can add the error values to the output of the program and plot them (along with the model and the data) in Matlab.

2. Add runtime arguments to your program so that the choices of input and output method are specified at run time. To convert the (string) arguments to integers, you can use the `atoi()` function (available if you `#include <stdlib.h>`). Once you have done this you can remove various lines of code from the program that handle getting these values from the keyboard.

3. Split code into two .c files: one containing the main function and the other containing the other functions (e.g. `splinefuns.c`). Create a .h file (e.g `splinefuns.h`). Create a makefile for your program and show that it works.