Level 5 Computational Physics — Exercise 1

The deadline for this exercise is **Sunday 21st October 2012** at midnight. Your report and all program (*.c) files should be uploaded into Blackboard at the appropriate point in the Computational Physics (PHYS2COMP) course.

S. Hanna

Objectives of the exercise

This starter exercise is designed to be straightforward. The purpose is to make sure that you have fulfilled the following requirements needed for the rest of the course:

- Become familiar with programming in C
- Understand how to enter, compile and run simple programs
- Understand the origins of numerical errors in computer programs
- Be able to produce output from a program and plot simple results graphically

Report

For each of the following problems, you should consider including the following:

- A brief statement of the problem as you understand it
- A brief description of the method used to solve the problem
- A presentation of your results, in graphical format where applicable
- An interpretation of the results, which should include a critique of the method and ideas for improvement.

As a guide, half to one page is probably sufficient for each section. Sometimes, you may need more space, e.g. if you have several graphs to display. Generally, you should aim for conciseness.

Problem 1: Quadratic Solver (6/20 marks)

Write a program to calculate the roots of a quadratic equation, $ax^2 + bx + c = 0$, using the usual formula. a, b and c should be entered by the user when given a suitable prompt. Test that your program is giving the right answer for a range of coefficients. Does it give the correct roots for simple equations where you already know the solution?

Because floating point numbers are stored with a finite number of bits, the answer from an arithmetic operation will usually have to be rounded before being stored. This rounding can cause errors in subsequent calculations involving that number, and poor algorithms will allow these errors to give significant discrepancies in the final result.

As an example, try your quadratic solver (using a 32-bit data type) for the equation $x^2 + 40x + 3 = 0$. Do the roots found satisfy the equation exactly? Comment on what you find.

Problem 2: Series expansion of an expression (6/20 marks)

It is often required to calculate a logarithmic or trigonometric expression in a computer program. All computer languages have built-in functions for this, which rely on series expansions.

To illustrate this, write a subroutine or function which uses the following Taylor expansion to calculate the value of the sin function:

$$\sin x = \sum_{n=0}^{N} \frac{(-1)^n}{(2n+1)!} x^{2n+1} \quad \text{where} \quad N \longrightarrow \infty$$

The subroutine or function should take the value of x, calculate a fixed number of terms N, and return the resulting value of the summed series. Test that your code produces a good approximation to the correct value of $\sin x$ for known points (e.g. $x = 0$, $x = \pi/2$, $x = -3\pi/2$).

Compare graphically (using a plotting program) the values returned by your function for a number of points over the range of the sin function, with the values returned by the built in sin() function in your language. How does the agreement compare as you vary N and the number of points evaluated? What is a reasonable value of N such that your evaluation is correct within the intrinsic error of the floating-point format?

The disagreement in value due to the finite value of N is an example of truncation error; the 'roughness' of the sin function calculated at just a few points is an example of discretisation error. Computers do not typically use the Taylor expansion method to calculate simple functions; there are much more efficient ways of evaluating simple functions numerically. You may wish to look into these and comment on them in your report.

Problem 3: Root-finding of a polynomial (8/20 marks)

Earlier, you wrote a program to find the roots of a quadratic equation. As noted in the lectures, higher-order polynomial equations (quintic and above) cannot be solved in this way, and so some form of numerical technique is required. In this problem, you will test the convergence of the Newton-Raphson method.

The Newton-Raphson approach relies on expanding the function $f(x)$ as a Taylor series and rearranging:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{1}$$

In general, each application of Eq. (1) yields an improved guess at the root, but we need to repeat the process iteratively until we have the required accuracy.

Write a program to find the roots of the quartic equation using the Newton-Raphson method:

$$f(x) = x^4 + x^3 - 12x^2 - 2x + 10 = 0$$

You should plot the function first to see where the zeros are. [Hint: they all lie in the range $[-4, 4]$]. In your report, you should indicate the initial guess required to obtain each root, and the numbers of iterations required to achieve accuracy of 6 significant figures.

Submitting your work

You should submit the following to Blackboard:

1. A concise report, in MS Word or pdf format;
2. Your final programs, as developed for problems 1, 2 and 3.

Please note the following points:

- Blackboard will not accept your compiled program—please only upload your "prog.c" files.
- Blackboard plagiarism checker (Turnitin) also won't accept a file ending ".c" so please rename your file with a ".txt" extension.
- Please also give your three programs sensible distinguishing names, including your name or userid e.g. "my_userid_ex1_prob_1.txt" or "myname_ex1_prob_3.txt".

If you have any problems submitting your work, please contact Dr. Hanna (s.hanna@bristol.ac.uk) or ask a demonstrator.