Spring, 2017

# Physics 6810: Assignment #3

These exercises are follow-ups to class and background-note discussion of Richardson extrapolation (session 4), eigensystems (session 5), adaptive numerical routines (the last one is a bonus problem), and converting GSL to class form (also a bonus). These involve modifications to the `derivative_test.cpp` code and the `eigen_basis.cpp` code. Please ask questions! The due date will be specified online.

Please use BuckeyeBox to "hand in" the assignment. Put your homework inside a subfolder of your main folder named `PS_3` or `Assignment3` or similar.

Include in your folder makefiles (one for each program), C++ programs (with the answers to any questions in the comments at the top), and postscript files of any plots (with plot files). Using gnuplot plot files is required. *It is required that your code have appropriate comments.* Comment your codes with your name, email, AND revision history, as in the example codes from class. *Check the 6810 webpage for suggestions and hints.* Please ask questions and give feedback early and often. NOTE: You must do one of the BONUS problems to get a plus.

1. It's time to start planning your 6810 project. Please send email (as soon as you can, separate from the rest of the homework) to furnstahl.1@osu.edu with a (brief!) description of your ideas for a project. They can be vague at this point; we will refine them over the next few weeks! See the 6810 webpage for some past project descriptions.

2. Add a subroutine to take the Richardson extrapolation used in the "`extrap_diff`" subroutine one step further. That is, `extrap_diff` calls `central_diff` with two different values of $h$ and then combines them to extrapolate to smaller h (leading to an error proportional to $h^4$). Now write a new routine (called `extrap_diff2`) that calls `extrap_diff` with two different values of $h$ and combines them appropriately to get a still steeper dependence of the error on $h$. Verify the result by making an error plot (you may want to increase the starting value of $h$ to 0.5). [A new version of `derivative_test.cpp` with `extrap_diff` explicitly written with `central_diff` is available from the hints.]

3. Modify `eigen_basis.cpp` so that you can print out (to a file) the approximate wave function corresponding to a given state (e.g., the ground state or the first excited state). Plot the exact ground state wave function and the approximate wave function (as a function of r) for one of the potentials (your choice; I like the Coulomb best!) with two choices for $b$ (your choice!), each for basis sizes of 1, 5, 10, and 20. Comment on the

nature of the convergence and speculate about choosing b based on your plots. Make sure that the wave functions are normalized.

4. (BONUS) Make the calculation using the central difference method *adaptive*. That is, you specify the function but don't specify the value of $h$. Instead, your program determines (or, more precisely, estimates) the optimal value of $h$ automagically and uses that. Compare the h chosen by your program for the function described above to the value you would select based on the error plot.

5. (BONUS) Devise a measure of how close the approximate wave function is to the exact wave function and determine how this measure scales with the basis size $D$.

6. (BONUS for experts) Create a class that wraps a GSL integration function (e.g., such as qags) and a test program (e.g., like `qags_test.cpp` from Session 4) to demonstrate how it works.