

Physics 780.20: Assignment #4

These exercises are follow-ups to class tasks using various GSL functions. Please ask questions! It is due at the end of the day (midnight) on Monday, March 6 (this is much more time than you need, but I will be out of the country the week of February 27). Note: Your projects will be due by midnight on Thursday, March 16.

To “hand in” the assignment, email me (at `furnstahl.1@osu.edu`) a “tarball” (see below) of your program and makefile (with the answers to any questions in the comments), a postscript file of the log-log plot, and a Mathematica notebook with the command(s) and result. (Send the tarball as an attachment.) Use C++ and any plotting program you want (gnuplot recommended for now). *Check the 780.20 webpage for suggestions and hints if you get stuck.* Please give feedback early and often.

To make a tarball, create a directory with the name: `your_name_ps4` (replace “your_name” by your last name!), e.g., `mkdir furnstahl_ps4`

Copy all of the files to pack up into that directory. Then give the command:

```
tar cvzf furnstahl_ps4.tarz furnstahl_ps4
```

and you’re done! The command “`tar tfvz furnstahl_ps4.tarz`” lists the contents, so you can check that you’ve packed all the files correctly. [Note: The “c” is for “create a new archive”, the “f” is for file and goes with the archive name `furnstahl_ps4.tarz`, “v” is for verbose mode (say what is going on), and “z” means to compress the tar file with gzip (tar stands for “tape archive”).]

1. Please send email to `furnstahl.1@osu.edu` by March 3 with a (brief!) progress report on your project. In particular, include a description of your project goal and a list of possible subgoals. Subgoals are things like: write a test code, convert a code from a different language, do an error analysis, etc. The project for 780.20 can be a piece of a bigger project (that doesn’t have to be completed this quarter) or can be self-contained. You are free to adjust the project goal and subgoals as you proceed, but I want to have a rough idea where you’re heading before it’s due!
2. **Cubic Splining.** Your goal is to carry out slightly modified versions of steps 3 and 4 of the Cubic Splining task from Session 9.
 - (a) Modify the `spline_function2` code so that it splines the ground-state hydrogen

wave function (in the units we've used before):

$$u(r) = 2r e^{-r} .$$

(You can delete the second spline in the code.)

- (b) Determine how many (equally spaced) points to use to represent the wave function. Suppose you need the derivative of the wave function to be accurate to 10^{-6} for $1 < r < 4$ (absolute, not relative error) Devise (and carry out!) a plan that will tell you the number of points needed to reach this goal. [Hint: Think about a graph you could make.]
3. **Nonlinear Least-Squares Fitting with GSL Routines.** Your goal is to carry out parts 4 and 5 of this task from Session 10 (see the handout). Be sure to answer all the questions in part 4. Many of you have already completed the first three parts of this task, which give you a good start on parts 4. and 5. If you haven't, go through them first.
4. (BONUS) Write a code that reads in the $x(t)$ vs. t data generated by `diffeq_oscillations.cpp` or `diffeq_pendulum.cpp` and finds the power spectrum. You can use the `fourier.c` code from Landau and Paez (available from the 780.20 web page in the "Computer Codes and Makefiles" section).