Pre-class assignment # 3

PHY-905-005
Computational Astrophysics and Astrostatistics
Spring 2023

This assignment is due the evening of Wednesday, January 18, 2023.

Instructions: Read the materials below and follow the instructions/answer the questions at the end. Turn in your code and all materials via the GitHub Classroom.

What to turn in: Turn in plots (and the scripts used to generate them), source code, etc. **Do not** turn in object files, binary files, or very large files of any kind unless you are explicitly asked to do so!

Reading:

1. Interpolation of data: Sections 2.1 and 2.4 of *An Introduction to Computational Physics*, by T. Pang (PDF; provided).
2. Root and extrema finding: Sections 3.3 and 3.4 of *An Introduction to Computational Physics*, by T. Pang (PDF; provided)
3. Wikipedia article on interpolation techniques (optional, but highly recommended)
4. Wikipedia article on root-finding techniques (optional, but highly recommended)
5. Interpolation: Section 5.11 of *Computational Physics*, by M. Newman (optional)
6. Root and extrema finding: Sections 6.3 and 6.4 of of *Computational Physics*, by M. Newman (optional)

Questions:

1. Implement the linear interpolation method as a function and test it on the arrays of x and $f(x)$ values generated by the file `interp.py` for values of x that are midway between the given data points. Compare the outputs of your linear interpolation to the actual analytic function (also given in `interp.py`). How good of a job does it do? Now, try doing this for the same values of x using the SciPy `scipy.interpolate` package – in particular, `interp1d`. `interp1d` lets you try different methods using the `kind` argument. Try the `linear`, `nearest`, and `cubic` methods. How well do these perform compared to the routine you've written?
2. Implement the bisection method and Newton's method for finding the roots of a mathematical function. Put each one into its own function with any arguments that are necessary. Test the two methods on the three functions in the file `root.py`, using guesses for your starting point that are both close to and far from the actual root. Some of these functions are better behaved than others for different root-finding methods; print out the values of x and $f(x)$ (and $f'(x)$ if relevant) as your code iterates. Describe the outcome in `ANSWERS.md`.
3. After you have done the required reading and implemented the code described above, what remaining questions do you have about these numerical techniques? List those in `ANSWERS.md`.