In-class assignment #7

PHY-905-005
Computational Astrophysics and Astrostatistics
Spring 2023

Instructions: We're going to use the code for your pre-class assignment, where you modeled the behavior of a mass on a spring over some interval of time. You've already implemented the Euler method and a simple predictor-corrector method. Now, implement (1) either the Euler-Cromer method or the *explicit* midpoint method and (2) the 4th order Runge-Kutta method (RK4), and answer the following questions:

- For given timestep sizes, $\Delta t = 0.1\pi, 0.01\pi$, and 0.001π, what is the difference in the relative change in energy between the Euler method and these two methods?
- Assume you wish to maintain energy to a given level of accuracy - say 0.01% between $t = 0$ and $t = 4\pi$. How many time steps of each of your methods do you need to reach that level of accuracy? How many total floating-point operations is that for each method for the entire integration? To attain a given level of accuracy, which is the least computationally expensive method to use?
- If you wanted to maintain energy to a given level of accuracy from time step to time step, how might you go about devising an algorithm to do it? In other words, how to you change the way that you do integration so that you can adapt your time steps to save computational cost?

Make some notes in `ANSWERS.md`, and we'll also discuss it in class.

Some hints/suggestions:

- A Python implementation of the 4th order Runge-Kutta method is in Section 8.1.3 of Newman – I suggest you adapt that rather than trying to create it from scratch!
- Make sure that you use the exact same times for each algorithm when plotting, otherwise you'll see "phase drift" in your errors!
- Write code as functions to the greatest extent possible – you'll be able to reuse it later in the class!

What to turn in: Turn in `ANSWERS.md`, any source code you wrote, any plots you created (and the scripts you used to create them). **Do not** turn in object files or executables!