PHY 604: Computational Methods in Physics and Astrophysics II

Homework #2

Due: Oct. 1, 2025

Programs can be written in any language (but python is recommended), In addition to the program, you should have a writeup that contains the plots requested in the homework questions, answers to any analytical or explanation questions, and a short description of your code and how to run it. This can be done in, e.g., LaTeX, markdown, etc. Combining the code and writeup in jupyter notebooks is highly recommended.

Code and writeup should be submitted using `git` *via github in the repo that was created from* github classroom *link.*

1. *A more stable version of Lagrange interpolation* Recall the equation for the Lagrange polynomial we used to interpolate (x_i, f_i) points:

$$p(x) = \sum_{i=0}^{n-1} l_i(x) f_i \tag{1}$$

where

$$l_i(x) = \prod_{j=0, i\neq j}^{n-1} \frac{x - x_j}{x_i - x_j}. \tag{2}$$

(a) Show that we can rewrite the polynomial in the so-called Barycentric form:

$$p(x) = \frac{\sum_{j=0}^{n} \frac{w_j}{x - x_j} f_j}{\sum_{j=0}^{n} \frac{w_j}{x - x_j}} \tag{3}$$

where w_j are weights given generally by $w_j = \frac{1}{\Pi_{k\neq j}(x_j - x_k)}$.

(b) Write a program that uses this Barycentric form to perform interpolation on the "Runge function":

$$f(x) = \frac{1}{1 + 25x^2} \tag{4}$$

in the range of $[-5, 5]$. Use the Chebyshev distribution of interpolating points discussed in class. Note that since the weights only depend on the distribution of points, they can be determined generally with no knowledge of the function or the final points we interpolate onto. For the Chebyshev distribution of the first kind, $w_j = (-1)^j \sin\left[\frac{(2j+1)\pi}{2n+2}\right]$. Also, you will need to be careful with the situation where the interpolated point x happens to coincide with one of the x_j; one of the strengths of the Barycentric version is that the weighting function appears both in the numerator and denominator, and thus the final result should not be too sensitive to exactly how you avoid the divergences.

(c) Compare the interpolation using the Barycentric form to the version using Eqs. 1 and 2 discussed in class. Specifically, test for the number of points $n \approx 1000$ and describe what you see.

2. *Cubic spline of a noisy function* Consider the function

$$f(x) = \frac{\sin(x - x^2)}{x}.$$

(a) Write a program that adds random noise to $f(x)$ so that $f_{\text{noisy}}(x) = f(x) + \lambda\delta(x)$, where $\delta(x)$ is a random number in the range $[-1, 1]$, and λ is a scaling factor. Plot $f(x)$ and $f_{\text{noisy}}(x)$ in the range from $[0.5, 10]$. Take as your data set for parts (b) and (c) 100 equally-spaced points of $f_{\text{noisy}}(x)$ in that range.

(**b**) Calculate the numerical derivative of this dataset across the range. By plotting $f'(x) - f'_{\text{noisy}}(x)$, comment on how the noise in the derivative changes with λ, i.e., compared to the level of noise in the data.

(**c**) Write a program that performs a cubic spline interpolation of the noisy data set (feel free to base it on the program discussed in class, but please do not just use a library). Use the spline to (analytically) take the first derivative of the data. Comment (using plots) on the noise in the numerical derivative in (b) versus what you obtain from the first derivative of the spline, and how it changes with λ.

3. *Multiple roots of functions* Write a program that will find all of the real roots in a given interval of a given function using the Bisection, Newton-Raphson, and Secant methods. Test your program on the function from the previous problem:

$$f(x) = \frac{\sin(x - x^2)}{x}.$$

Comment on the relative speed of convergence of the Bisection method versus the Newton-Raphson method.

4. *Three-Body Problem.* (this is basically Newman Exercise 8.16) Consider three stars, initially at rest, with masses and initial positions: All stars are in the x-y plane, with $z = 0$. The equation of motion

star	**mass**	x	y
1	150	3	1
2	200	−1	−2
3	250	−1	1

for star i is then

$$\frac{d^2\mathbf{x}_i}{dt^2} = \sum_{j=1;j\neq i}^{3} G m_j \frac{\mathbf{x}_j - \mathbf{x}_i}{|\mathbf{x}_j - \mathbf{x}_i|^3} \tag{5}$$

where $i = 1, 2, 3$, and $\mathbf{x}_i = (x_i, y_i)$ are the coordinates of star i.

Write this as a system of first-order ODEs by introducing the velocity, $\mathbf{v}_i = d\mathbf{x}_i/dt$. Work with $G = 1$.

(**a**) Solve this system from $t = 0$ to $t = 2$ using the *adaptive* 4th order Runge-Kutta method, seeking a local error of $\epsilon = 10^{-5}$.

Note: you may want to prevent yourself from dividing by zero if the stars get on top of one-another by introducing some small safety factor in the denominator of the force expression. In N-body calculations, this is called a *softening length.*

Make a plot of the trajectories of the 3 stars.

(**b**) The center of mass of the system should remain fixed. Compute the center of mass as:

$$x_{\text{cm}} = \frac{1}{M}\sum_{i=1}^{3} m_i x_i \tag{6}$$

$$y_{\text{cm}} = \frac{1}{M}\sum_{i=1}^{3} m_i y_i \tag{7}$$

where M is the sum of masses.

Plot the value of the center of mass vs. time (e.g. x_{cm} vs. t, and y_{cm} vs. t).