PHY 604: Computational Methods in Physics and Astrophysics II

Homework #1

Due: 09/17/2025

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. *Understanding round-off error*: *Note, no program is required for this question.* Consider a quadratic equation of the form $ax^2 + bx + c = 0$. The two solutions of this are:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{1}$$

(**a**) Explain how this expression may be problematic with respect to roundoff errors if b is much larger than a and c. Recall that such errors often occur when subtracting close large numbers.

(**b**) Provide an alternative expression that will have smaller errors in the situation you describe in **(a)**. ***Hint:*** The same trick should work as for the "Round-off error example" in Lecture 1.

2. *Round-off error and accurate calculation of the exponential series*: Consider the series expansion for an exponential function:

$$e^x \simeq S_n(x) = 1 + \frac{x}{1!} + \frac{x^2}{2!} + ... + \frac{x^n}{n!}. \tag{2}$$

(**a**) Write a program that computes the exponential function using this series expansion for given number of terms n.

(**b**) For n ranging between 0 and 100, compare the result with the exponent calculated with a built-in function or function form a numerical library (e.g., `numpy.exp`) in the following way. Plot the error defined by

$$\epsilon_n = \frac{|e^x - S_n(x)|}{e^x} \tag{3}$$

on a log-log plot for a large positive and large negative exponent (e.g., $x = 20$ and $x = -20$). Describe what you see.

(**c**) Consider the following (trivial) equality: $e^{-x} = \left(e^{-1}\right)^x$. Write a program that utilizes this equality to get a more accurate series expansion for large negative exponents. Plot ϵ_n on a log-log plot to demonstrate that you have achieved this.

3. *Errors in numerical differentiation*: Calculate the derivative of the function $f(x) = \sin(x)$ at the point $x = \pi/4$ using the first-order forward difference. Plot on a log-log plot the error with respect to the analytical derivative for a wide range of Δx. Describe the behavior you see (especially for very small Δx) and the reason for the trends. How does it change if you use a second-order central difference? How about a fourth-order central difference?

4. *Comparing methods of integration*: (based on Newman exercise 5.7) Consider the function:

$$I = \int_0^1 \sin^2\left(\sqrt{100x}\right) dx. \tag{4}$$

(**a**) Plot the integrand over the range of the integral.

(**b**) Write a program that uses the *adaptive trapezoid rule* to calculate the integral to an approximate accuracy of $\epsilon = 10^{-6}$, using the following procedure: Start with the trapezoid rule using a single subinterval. Double the number of subintervals and recalculate the integral. Continue to double the number of subintervals until the error is less than 10^{-6}. Recall that the error is given by $\epsilon_i = \frac{1}{3}(I_i - I_{i-1})$ where the number of subintervals N_i used to calculate I_i is twice that used to calculate I_{i-1}. To make your implementation more efficient, use the fact that

$$I_i = \frac{1}{2} I_{i-1} + h_i \sum_k f(a + kh_i) \tag{5}$$

where h_i is the width of the subinterval for the ith iteration, and k runs over *odd numbers* from 1 to $N_i - 1$.

(**c**) Write a separate program that uses *Romberg integration* to solve the integral, also to an accuracy of 10^{-6} using the following procedure. First calculate the integral with the trapezoid rule for 1 subinterval [as you did in part (b)]; we will refer to this as step $i = 1$, and the result as $I_1 \equiv R_{1,1}$. Then, calculate $I_2 \equiv R_{2,1}$ using 2 subintervals (make use of Eq. 5). Using these two results, we can construct an improved estimate of the integral as: $R_{2,2} = R_{2,1} + \frac{1}{3}(R_{2,1} - R_{1,1})$. In general

$$R_{i,m+1} = R_{i,m} + \frac{1}{4^m - 1}(R_{i,m} - R_{i-1,m}). \tag{6}$$

Therefore, for each iteration i (where we double the number of subintervals), we can obtain improved approximations up to $m = i - 1$ with very minor extra work. For each i and m, we can calculate the error at previous steps as

$$\epsilon_{i,m} = \frac{1}{4^m - 1}(R_{i,m} - R_{i-1,m}). \tag{7}$$

Use Eqs. 6 and 7, to iterate until the error in $R_{i,i}$ is less than 10^{-6}. How significant is the improvement with respect to number of subintervals necessary compared to the approach of part (b)?

(**d**) Use the Gauss-Legendre approach to calculate the integral. What order (i.e., how many points) do you need to obtain an accuracy below 10^{-6}? You can find tabulated weights and points online, e.g.,

https://pomax.github.io/bezierinfo/legendre-gauss.html.

5. *Integration to* ∞: (based on Newman). Consider the gamma function,

$$\Gamma(a) = \int_0^\infty x^{a-1} e^{-x} dx \tag{8}$$

We want to evaluate this numerically, **and we will focus on** $a > 1$. Consider a variable transformation of the form:

$$z = \frac{x}{x + c} \tag{9}$$

This will map $x \in [0, \infty)$ to $z \in [0, 1]$, allowing us to do this integral numerically in terms of z.

For convenience, we express the integrand as $\phi(x) = x^{a-1} e^{-x}$.

(**a**) Plot $\phi(x)$ for $a = 2, 3, 4$.

(**b**) For what value of x is the integrand $\phi(x)$ maximum?

(c) Choose the value c in our transformation such that the peak of the integrand occurs at $z = 1/2$—what value is c?

This choice spreads the interesting regions of integrand over the domain $z \in [0, 1]$, making our numerical integration more accurate.

(d) Find $\Gamma(a)$ for a few different value of $a > 1$ using and numerical integration method you wish, integrating from $z = 0$ to $z = 1$. Keep the number of points in your quadrature to a reasonable amount ($N \lesssim 50$).

Don't forget to include the factors you pick up when changing dx to dz.

Note that roundoff error may come into play in the integrand. Recognizing that you can write $x^{a-1} = e^{(a-1)\ln x}$ can help minimize this.