PHYS 305: Computational Physics II

Winter 2018

Homework #1

(Due: January 29, 2018)

Each problem is worth 10 points. E-mail your soultions to `steve@physics.drexel.edu` *with a subject including PHYS 305 and the Homework number. The e-mail should have as an attachment a zip (or tar) file containing a PDF document containing all discussion, reasults, and graphs requested, and files containing Python scripts for all programs written.*

1. (a) Write a script to compute the derivative of the function $f(x) = x^2\, e^{-x}$ at the point $x_0 = 3$, using both the one-sided

$$f_p^{(1)} = \frac{f(x_0 + \delta x) - f(x_0)}{\delta x}$$

and centered

$$f_p^{(2)} = \frac{f(x_0 + \delta x) - f(x_0 - \delta x)}{2\delta x}$$

approximations. As in class, compare the errors made by the two methods by plotting

$$e^{(1)} = |f_p^{(1)} - f'(x_0)|$$

and

$$e^{(2)} = |f_p^{(2)} - f'(x_0)|$$

against δx on a clearly labeled log–log plot, for $\delta x = 2^{-n}$, with $n = 1, 2, 3, \ldots 50$.

(b) Now explore how the derivative can be obtained by extrapolating the numerical results to $\delta x = 0$. Evaluate $f_p^{(2)}$ for $\delta x = 0.1, 0.05, \text{and } 0.025$, and save the result in a `numpy` array `deriv`. Use the `numpy` `polyfit` function to fit the array as a quadratic in δx (also saved as an array) of the form

$$e^{(2)} = p_0\, \delta x^2 + p_1\, \delta x + p_2.$$

The array of coefficients `p` is returned by the `polyfit` call:

```
p = polyfit(dx, deriv, 2)
```

Print out the coefficients p_i and hence determine the extrapolated derivative for $\delta x = 0$ and its difference from the true value. How does this error compare with the error obtained using the smallest δx in the array?

(c) Repeat part (b) using a cubic fit to the points obtained with $\delta x = 0.1, 0.05, 0.025, \text{and } 0.0125$.

2. (a) Write a script that computes

$$I = \int_0^2 x \cos x \, dx$$

using (i) the basic integration scheme described in class, (ii) the trapezoid rule, and (iii) Simpson's rule (see Numerical Recipes, Sec. 4.1, 4.3 for details). Evaluate the integral using each scheme with N intervals between $x = 0$ and $x = 2$, for $N = 4, 8, 16, \ldots 2^{20}$, and plot the errors versus $\delta x = 2/N$ on a log–log plot.

(b) Evaluate the integral using the trapezoid rule for $N = 4, 8, 16,$ and 32, fit the result with a cubic polynomial in δx, as in question 1, and hence determine the extrapolated integral for $\delta x = 0$. Again calculate the error, and compare it with the error obtained using just $\delta x = 1/16$.

3. (a) A particle moves in an almost inverse-square potential with

$$\phi(r) = -\frac{GM}{(r^2 + \varepsilon^2)^{1/2}},$$

where $GM = 1$ and $\varepsilon = 0.1$. Following the development in class, we can write

$$\begin{aligned} E &= \tfrac{1}{2}(v_r^2 + v_t^2) + \phi(r) \\ L &= r v_t \end{aligned}$$

so

$$\begin{aligned} E &= \tfrac{1}{2}v_r^2 + \frac{L^2}{2r^2} + \phi(r) \\ v_r^2 &= 2[E - \phi(r)] - \frac{L^2}{r^2}. \end{aligned}$$

(i) Use bisection (see Numerical Recipes, Sec. 9.1) or a built-in Python root finder to determine the two turning points of the orbit—that is, the values $r = r_\pm$ where $v_r^2 = 0$—for $E = -0.5$ and $L = 0.5$.

(ii) The period of the orbit is

$$P = 2\int_{r_-}^{r_+} \frac{dr}{v_r}.$$

Note that the integrand is (integrably) singular at the end points, making a Newton–Cotes rule like trapezoid unusable without modification. Near $r_\pm$, v_r^2 goes linearly to zero, so we can write $v_r^{-1} = (r - r_-)^{-1/2}(r_+ - r)^{-1/2} f(r)$, where the function

$$f(r) = 2\,\frac{(r - r_-)^{1/2}(r_+ - r)^{1/2}}{v_r}$$

is regular and positive over the entire range.

As discussed in class, the Gauss–Chebyshev quadrature formula is designed to handle integrands with precisely this behavor. Evaluate the integral using the quadrature

$$P \approx \frac{r_+ - r_-}{2} \sum_{i=0}^{n-1} w_i \, f(r_i)$$

with $n = 10$ points, where

$$\begin{aligned} r_i &= \frac{r_+ + r_-}{2} + \frac{r_+ - r_-}{2} x_i \\ x_i &= \cos\left[\frac{(2i+1)\pi}{2n}\right] \\ w_i &= \frac{\pi}{n}, \end{aligned}$$

for $i = 0, \ldots n-1$ (see Numerical Recipes, Sec. 4.5 and the file `gaus_cheb.py` on the course web site).

(b) Repeat the calculation for the case $\varepsilon = 0$, with the same E and L as in part (a), and compare your answer to the Newtonian result

$$P = 2\pi GM\,(-2E)^{-3/2}.$$