Graduate Computational Physics Problem Set #4
Prithivi Rana. (09/26)

Problem 1. *Exercise 5.9 of Newman: Heat capacity of a solid*

Solution. This problem tasks us with writing a function to calculate the specific heat capacity (Cv) for a given temperature of solid aluminum and plotting a graph of the values over a range of temperatures. The specific heat capacity can be calculated using the following equation:

$$C_v = 9V\rho k_B \left(\frac{T}{\theta_D}\right)^3 \int_0^{\theta_D/T} \frac{x^4 e^x}{(e^x-1)^2} dx \tag{1}$$

where, `V` is the volume of the solid, ρ is the number density of atoms, k_B is the Boltzmann's constant, and θ_D is the Debye temperature.

a) For the first part of the problem, I wrote a function `cv(T)` that calculates the C_V for a given temperature. The function name and input parameters were defined as:

```
def cv(T= None, V= 1000e-6, rho= 6.022e28, thetad= 428, N= 50)
```

where the volume(V), density(rho), and Debye temperature(thetad) have default values for aluminum given in the problem definition (note: volume has been converted to m^{-3}) unless different values are specified by the user for any other material. The N represents the number of sample points for the Gaussian quadrature. It will be set to 50 for parts a and b of the problem.

The function then calculates the values of the coefficients first and then performs a Gaussian quadrature for the integrand in equation 1 over the range [0,θ_D/T] for a temperature (T) specified by the user. The SciPy function `integrate.fixed_quad` was used to evaluate the integral with Gaussian quadrature in this function.

b) For part b, I used the function `cv(T)` for a range of temperatures from 5K to 500 K at differences of 0.1 K. This data is plotted in figure 1 which shows the heat capacity as a function of temperature for Aluminum.

c) Part c tasked us with testing the convergence by evaluating choices for N=10, 20, 30, 40, 50, 60, 70.

I first started off by repeating the process done in part b for different values of n. This gave me seven curves of Cv vs T, evaluated using Gaussian quadrature with different numbers of sample points, which is plotted in figure 2. Just plotting them with the regular axes scales proved to be useless in seeing their convergence since the differences between the multiple evaluations were too small. So, all the curves just overlapped each other and appeared as a single curve in the plot.

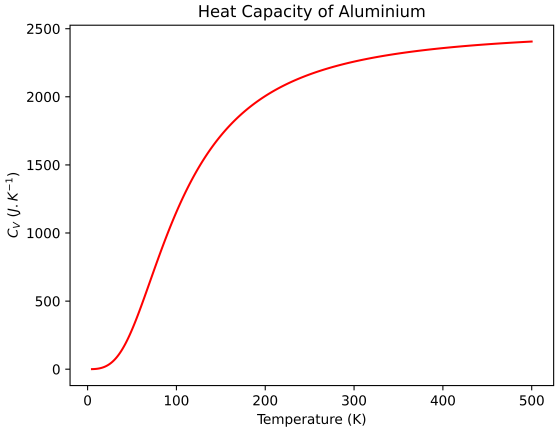

Figure 1: C_V of Aluminum

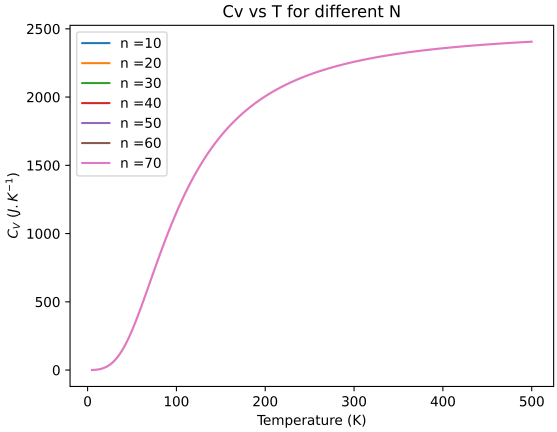

Figure 2: C_v of Aluminum computed with different n parameters for Gaussian Quadrature

In order to better visualize the convergence, I calculated the difference between the results of a particular value of n and the resulting curve from Gaussian quadrature with n=70, since that was the highest value. I then plotted the relative differences using a log scale for both axes which resulted in figure 3. The figure gives a better visual representation of the convergence of the curves towards higher values.

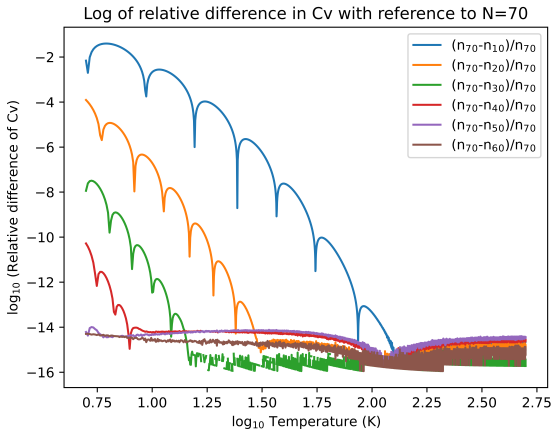

Figure 3: Relative difference between C_Vs computed with different n parameters for Gaussian Quadrature

Problem 2. *Read Example 5.10 in Newman. Period of an anharmonic oscillator*

Solution.

Problem 2 tasks us with analyzing the period of an anharmonic oscillator. We start off with the equation for the total energy as the sum of kinetic and potential energies.

$$E = \frac{1}{2}m\left(\frac{dx}{dt}\right)^2 + V(x) \tag{2}$$

where, `x` is the position of the particle, `m` is the mass, and `V(x)` is the potential energy.

a) Part a asks us to derive an equation for the period(T).

The problem assumes that the potential `V(x)` is symmetric about x = 0. It is released from its amplitude i.e. `x = a` at `t=0` and it swings towards the origin. This tells us that, at `t = 0`, $\frac{dx}{dt} = 0$ which gives us the value of E:

$$E = V(a)$$

Substituting this in equation 2 and rearranging for $\frac{dx}{dt}$ is presented in the following lines:

$$
\begin{aligned}
V(a) &= \frac{1}{2}m\left(\frac{dx}{dt}\right)^2 + V(x) \\
\frac{1}{2}m\left(\frac{dx}{dt}\right)^2 &= V(a) - V(x) \\
\left(\frac{dx}{dt}\right)^2 &= \frac{2}{m}(V(a) - V(x)) \\
\frac{dx}{dt} &= \sqrt{\frac{2}{m}(V(a) - V(x))}
\end{aligned}
$$

Once we have the equation in this form, we can integrate both sides with respect to t from 0 to $\frac{1}{4}$T:

$$
\begin{aligned}
dt &= \sqrt{\frac{m}{2(V(a) - V(x))}}dx \\
\int_0^{\frac{1}{4}T} dt &= \sqrt{\frac{m}{2}}\int_a^0 \frac{1}{\sqrt{V(a) - V(x)}}dx \\
\frac{1}{4}T &= \sqrt{\frac{m}{2}}\int_a^0 \frac{1}{\sqrt{V(a) - V(x)}}dx \\
T &= \sqrt{\frac{16m}{2}}\int_a^0 \frac{1}{\sqrt{V(a) - V(x)}}dx
\end{aligned}
$$

or,

$$
T = \sqrt{8m}\int_0^a \frac{1}{\sqrt{V(a) - V(x)}}dx \tag{3}
$$

b) As the problem stated, we set `V(x)` $= x^4$ and `m` $= 1$. I wrote a function that uses SciPy's `integrate.fixed_quad()` function with n=20 and evaluated Equation 3 for amplitudes(`a`) over the range [0,2]. This gives us the graph in figure 4.

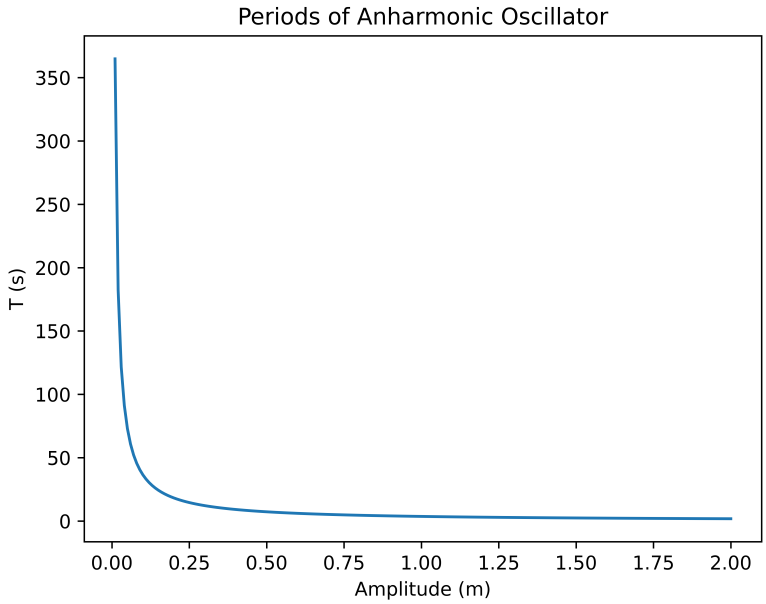

Figure 4: Amplitude vs Period for Anharmonic Oscillator

c) Figure 4 shows us that the period decreases for bigger amplitudes. This means that the oscillator is getting faster as the amplitude increases, even though it is traveling a longer distance. This is because the anharmonic oscillator corresponds to a quadratic potential; $V(x) \quad \alpha \quad x^2$. So, the restoring force increases with a greater magnitude for bigger amplitudes. This in turn increases the acceleration and velocity even though the particle has to travel further.

We also see that the period diverges as the amplitude goes to zero. This is also because of the quadratic potential. As the position goes towards zero, the restoring force becomes small, making the acceleration and velocity extremely small as well.

Problem 3. *Exercise 5.13 in Newman. Then: (d) Perform the calculation using Gauss-Hermite quadrature (scipy can give you the right roots and weights to use). Can you make an exact evaluation (meaning zero approximation error) of the integral?*

Solution. This problem tasks us with plotting the wavefunction for different energy levels of a 1D quantum harmonic oscillator and also calculating the quantum uncertainty in the position of a particle.

The equation for the wavefunction is given by:

$$\psi_n(x) = \frac{1}{\sqrt{2^n n! \sqrt{\pi}}} \exp^{-x^2/2} H_n(x)$$

where n is the energy level and ranges from 0....inf, and H_n(x) is the nth Hermite polynomial.

a) For part a, the problem asks us to write a function `H(n,x)` that calculates the nth Hermite polynomial for a user-defined x and n ≥ 0. For this, we use the relation of Hermite polynomials:

$$H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x)$$

Since we are looking for the nth Hermite polynomial, I reworked the equation by changing the indices.

$$H_n(x) = 2xH_{n-1}(x) - 2(n-1)H_{n-2}(x)$$

I then wrote a recursive function that calculates the nth Hermite polynomial. The function calls itself recursively and terminates once n reaches 0. The function implementation is written below:

```
def H(n=None, x=None):

    if (n<0):
        raise ValueError("N cannot be negative")
    if (n==0):        #H_0(x) = 1
        return 1
    elif (n==1):      #H_1(x) = 2x
        return (2*x)

    return ((2*x*H(n-1,x)) - (2*(n-1)*H(n-2,x)))
```

Once I had a function that calculates the Hermite polynomial, I wrote a function `wave_func(x,n)` to calculate the wavefunction. This function takes inputs for x and n and returns the wavefunction. I used this to make a plot for harmonic oscillator wavefunctions for the 0,1,2, and 3 energy levels for x in the range of [-4,4]. This plot can be seen in figure 5.

b) For part b, I used the same function to make a plot of wavefunction for n=30 and x in the range [-10,10]. Since we are using only one n, I passed an array of x values to the `wave_func(x,n)` function with the call `wave_func(x_values,30`. Since this performs the calculation over the entire array without the need for a loop, the calculation was executed in a few seconds and it returned values for ψ over the entire range of x. This is plotted in figure 6.

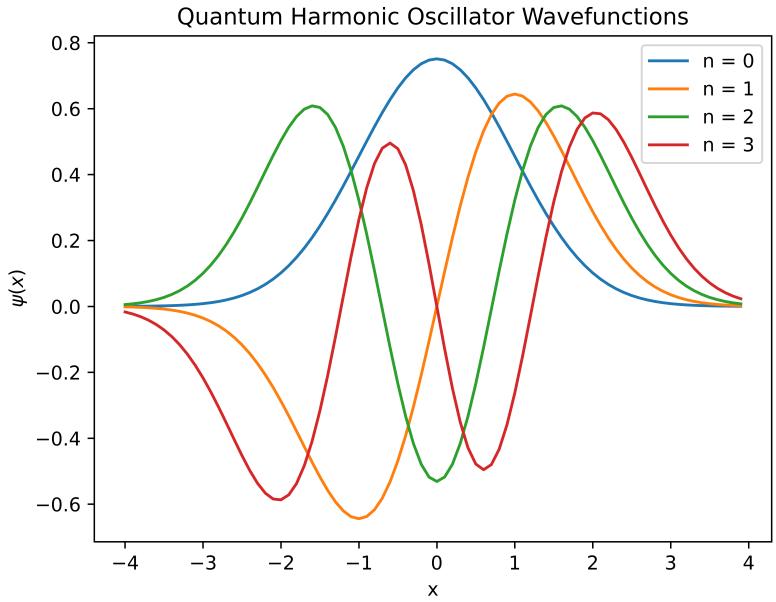

Figure 5: Wavefunctions for different energy levels

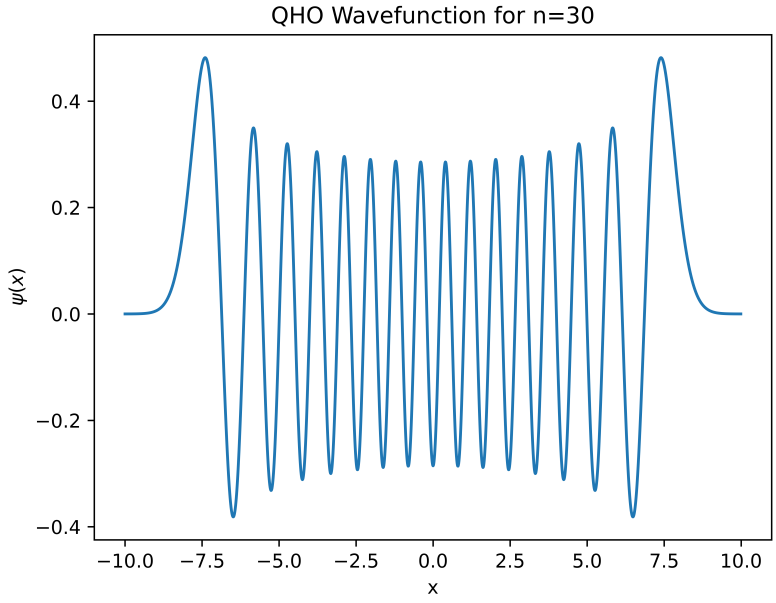

Figure 6: Wavefunction for 30th energy level

c) Part c asks us to calculate the quantum uncertainty in the position of a particle. We use the equation:

$$\langle x^2 \rangle = \int_{-\infty}^{\infty} x^2 |\psi_n(x)|^2 dx$$

where, the root mean square position, $\sqrt{\langle x^2 \rangle}$, quantifies the quantum uncertainty.

Although this can be evaluated using Gaussian quadrature, the limits of the integral go from $-\inf$ *to* $\inf$, so the integral needs to be rescaled first. For this, I used *5.72* from Newman's *Computational Physics* to find change of variables to rescale an integral from $-\infty$ *to* ∞ to -1 to +1:

$$x = \frac{z}{1-z^2}, \qquad dx = \frac{1+z^2}{(1-z^2)^2} dz$$

which transforms our integral:

$$\langle x^2 \rangle = \int_{-\infty}^{\infty} x^2 |\psi_n(x)|^2 dx = \int_{-1}^{+1} \frac{(1+z^2)z^2}{(1-z^2)^4} \left| \psi_n \left(\frac{z}{1-z^2} \right) \right|^2 dz$$

I implemented this in my program as the following rescaling function:

Listing 1: Function to rescale integrals over infinite ranges

```
def func_rescale(z=None):
    #function that rescales integral limits
    # -inf.... inf ---> -1...+1

    x = z/(1-z**2)
    dx = (1+z**2)/(1-z**2)**2
    return (dx * integrand(x=x))
```

Once I had the rescaling function written, I used SciPy's `integrate.fixed_quad()` function with n=100 over the range [-1,1] on our new integrand. Taking the square root of this value gives us an uncertainty for the 5th energy level, which was calculated to be:

$\sqrt{\langle x^2 \rangle} = 2.345207873785794.$

This is in the vicinity of $\sqrt{\langle x^2 \rangle} = 2.3$ as stated by the problem.

d) For part d, I used SciPy's `special.roots_hermite()`function with n=100. This gives us the roots and weights to perform the Gauss-Hermite quadrature. Before we compute the integral, we need to define `W(x)` and `f(x)`. Since `W(x)` $= \exp(-x^2)$ yields Gauss-Hermite quadrature, defining `f(x)` $= \exp(-x^2)\langle x^2 \rangle$ gives us our appropriate integrand.

Computing this integral and taking its square root gives us uncertainty for n=5 again with:

$\sqrt{\langle x^2 \rangle} = 2.345207879911713.$

I believe that an exact solution could only be possible if we have an exact known integrand for the function over the given limits. Gaussian quadrature and Gauss-Hermite quadrature gives us an answer that is close but even with a large number of sample points, it would still be an approximation. So, numerically it would be not be possible to make a zero approximation error of the integral.

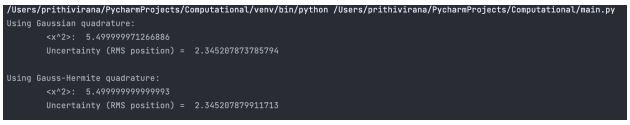

Figure 7: Root Mean Squared Positions from two integral methods