AMATH 481 & 581 Fall 2019

Homework 2 - Quantum Harmonic Oscillator/Boundary Value Problems

Submission open open until 01:00 AM Thursday October 24, 2019
The probability density evolution in a one-dimensional harmonic trapping potential is governed by the partial differential equation:

$$i\hbar\psi_t + \frac{\hbar^2}{2m}\psi_{xx} - V(x)\psi = 0, \tag{1}$$

where ψ is the probability density and $V(x) = kx^2/2$ is the harmonic confining potential. A typical solution technique for this problem is to assume a solution of the form

$$\psi(x,t) = \sum_{n=1}^{N} a_n \phi_n(x) \exp\left(-i\frac{E_n}{2\hbar}t\right), \tag{2}$$

and is called an eigenfunction expansion solution (ϕ_n=eigenfunction, $E_n > 0$=eigenvalue). Plugging-in this ansatz into Eq. (1) gives the boundary value problem

$$\frac{d^2\phi_n}{dx^2} - [Kx^2 - \varepsilon_n]\phi_n = 0 \tag{3}$$

where we expect the solution $\phi_n(x) \to 0$ as $x \to \pm\infty$ and $\varepsilon_n > 0$ is the quantum energy. Note that $K = km/\hbar^2$ and $\varepsilon_n = E_n m/\hbar^2$.
In what follows, take $K = 1$ and always normalize so that $\int_{-\infty}^{\infty} |\phi_n|^2 dx = 1$. You can do it in MATLAB by using a trapezoidal approximation of the integral:

$$\phi_{normalized} = \phi/\sqrt{(}\texttt{trapz}(x_{span}, \phi^2)))$$

See MATLAB documentation on `trapz` function for more details.

Calculate the first **five normalized** eigenfunctions (ϕ_n) and eigenvalues (ε_n) (up to tolerance of 10^{-4}) in increasing order such that the **first eigenvalue is the lowest one** using a shooting scheme. For this calculation, use $x \in [-L, L]$ with $L = 4$ and choose $xspan = -L : 0.1 : L$. Save the **absolute value** of the eigenfunctions in column vectors (vector 1 is ϕ_1, vector 2 is ϕ_2 and so on) and the eigenvalues in a separate **5x1 vector**.

Hint: Derive the boundary conditions at ±L as if these are the infinite boundaries, i.e. replacing $x = \pm\infty$ with $x = \pm L$ and performing the derivation used in the course notes, pages 22-23. Start with initial guess for the solution at $x = -L$ as $y(-L) = 1$.

ANSWERS: Should be written out as A1.dat–A5.dat for the eigenfunctions and A6.dat for the eigenvalues.

(2) Calculate the first **five normalized** eigenfunctions (ϕ_n) and eigenvalues (ε_n) in increasing order such that the **first eigenvalue is the lowest one** using the **direct method**. For this calculation, use $x \in [-L, L]$ with $L = 4$ and choose $xspan = -L : 0.1 : L$. Save the **absolute**

value of the eigenfunctions in column vectors (vector 1 is ϕ_1, vector 2 is ϕ_2 and so on) and the eigenvalues in a separate **5x1 vector**.

<u>**Hint 1:**</u> Formulate the harmonic oscillator problem as a differential e. value problem, i.e.,

$$\left[-\frac{d^2}{dx^2} + Kx^2\right]\phi_n = \varepsilon_n \phi_n \tag{4}$$

and discretize it using 2nd order central difference for interior points (without first and last points) to receive an e. value problem $A\vec{\phi_n} = \varepsilon_n \vec{\phi_n}$ where $\vec{\phi_n} = [\phi_n(x_2), ..., \phi(x_{N-1})]$. Such problems can be solved in MATLAB using the `eig` command.

<u>Hint 2:</u> Use a **bootstrap** approach to determine the boundary equations: To construct the matrix A use the derived boundary conditions (from question 1) and approximate the first and last points using 2nd order forward or backward difference and assume that Δx is small such that $\Delta x\sqrt{KL^2 - \varepsilon_n} \approx \ 0$. After you found the values of ϕ_n in the interior do not forget to compute the first and last points ($\phi_n(x_1)$ and $\phi_n(x_N)$) using full forward or backward-difference approximation (without assuming $\Delta x\sqrt{KL^2 - \varepsilon_n} \approx \ 0$). Be sure to save the eigenvectors including the first and last points, i.e., $\vec{\phi_n} = [\phi_n(x_1)\ \phi_n(x_2)\ , ...,\ \phi(x_{N-1})\ \phi_n(x_N)]$.

ANSWERS: Should be written out as A7.dat–A11.dat for the eigenfunctions and A12.dat for the eigenvalues.

(3) Looking back at problem 1, that you solved using the method of shooting, it is possible to recast this problem as an optimization problem, where we optimize over the eigenvalue starting with an initial guess.

Using the same initial guesses you used for problem 1, solve for the exact same eigenvalues again using an optimization approach through the function `fminunc`. You can use default options for this function.

For each eigenvalue computation, use the same guess that you used to start the shooting method. When it comes time to build the objective, use the same idea as in the shooting method, i.e. convert the 'check' you did for shooting into an error you try to minimize.

`fminunc` can be made to reveal the number of iterations it uses for each computation. Save the number of iterations you needed for each eigevnalue computation, so you have another **5x1 vector**.

ANSWERS: Should be written out as A12.dat for the eigenvalues and A13.dat for the iteration counts used to obtain these eigenvalues.