Exercise 5: *Numerical computation of time independent quantum harmonic oscillator eigenvalues and eigenfunctions*

Iriarte Delfina (1231682)

Quantum Information

November 17, 2020

Abstract

In this report, the **time independent quantum harmonic oscillator** in one dimension is study. The first eigenvalues and eigenfunctions are computed using the library LAPACK and compare with the theoretical ones. This is achieve by considering a **discrete** version of the Hamiltonian by means of the *Finite Difference Method.*

1 Introduction

In this project, the uni-dimensional quantum harmonic oscillator subject to a potential $V(x) = \frac{1}{2}m\omega^2 x^2$ is study. The Hamiltonian operator is defined as:

$$\hat{\mathcal{H}} = \frac{\hat{P}^2}{2m} + \frac{m\omega^2 \hat{X}^2}{2} \tag{1}$$

In particular, the quantum h.o. is a model that describes systems with a characteristic energy spectrum, given by a ladder of **evenly spaced energy levels**. The number of levels is infinite, but there must exist a minimum energy, since the energy must always be positive. Given this spectrum, we expect the Hamiltonian will have the form:

$$\mathcal{H}|n\rangle = (n + \frac{1}{2})\hbar\omega|n\rangle \tag{2}$$

where each level in the ladder is identified by a number n. As with any eigenvalue problem, we need an initial representation in which to work. In the $|x\rangle$ representation, the problem we need to solved is the differential equation:

$$-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2}\Psi_n(x) + \frac{1}{2}m\omega^2\Psi_n(x) = E_n\Psi_n(x) \tag{3}$$

for the eigenfunction $\Psi_n(x) = \langle x|\Psi_n\rangle$. Hence we have a **discrete spectrum** and the eigenstates of the harmonic potential must be **bound states** since the potential coming from the oscillator becomes infinite as $|x| \to \infty$ and therefore, the wavefunction must go to zero at large distances from the origin in order for the energy of the system to remain finite. The wavefunction solution are (in the $|x\rangle$-representation):

$$\Psi_n(x) = \frac{1}{\sqrt{n!2^n}}(\frac{m\omega}{\pi\hbar})^{1/4}\mathbb{H}_n(\zeta)e^{-}-\frac{\zeta^2}{2} \qquad \zeta = \sqrt{\frac{m\omega}{\hbar}}x \tag{4}$$

To solve numerically this equation we used the **Finite Difference Method** which consist on introducing a lattice in the space (dx) to define a discrete eigenproblem. We denote the wave function at point x_k as ψ_k. In particular, the second derivative can be discretize as follows:

$$\frac{d^2\psi}{dx^2}(x_k) = \frac{\psi_{k+1} - 2\psi_k + \psi_{k-1}}{(dx)^2} + o(dx^2) \tag{5}$$

The **time-independet Schrodinger equation at the fourth order** in one dimensional interval can then by written as:

$$\mathcal{H} = \begin{pmatrix} \frac{2\hbar^2}{2mdx^2} + \frac{1}{2}m\omega^2 x_1^2 & -\frac{\hbar^2}{2mdx^2} & 0 & \cdots & 0 \\ -\frac{\hbar^2}{2mdx^2} & \frac{2\hbar^2}{2mdx^2} + \frac{1}{2}m\omega^2 x_2^2 & -\frac{\hbar^2}{2mdx^2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & -\frac{\hbar^2}{2mdx^2} & \frac{2\hbar^2}{2mdx^2} + \frac{1}{2}m\omega^2 x_N^2 \end{pmatrix} \tag{6}$$

In this project, the first energies and wavefunction of the harmonic oscillator are solved by founding the correspondent eigenvalues and eigenfunction of the matrix 6 using the library LAPACK in FORTRAN.

2 Code development

The first step of the code development is to build the mesh, this is done by taking as input the number of points `n_point` desire and the size of the simulation box `L`, which then would be provided by an automated script in PYTHON. The potential generation is shown in Listing 1 which is nothing else as a typical harmonic oscillator potential centred in 0.

```
dx = 2._pr *L/dble(n_points-1) !Step mesh
do i = 1, n_points !potential
    x(i) =-L + ( float(i) - 1) * dx
    v(i) = 0.5_pr * mass *  omega *omega* x(i) * x(i)
end do
```

Listing 1. Generation of the harmonic potential in FORTRAN

The matrix generation in FORTRAN, given by Matrix 6, is shown in Listing 2. After initializing the matrix, the same subroutine built for the previous week Exercise is implemented in order to compute the eigenfunctions and eigenvalues. The `zheev` function from LAPACK is called with the first parameter of the function given as `'V'` in order to obtain both, the eigenvalues and eigenfunctions. The computed eigenfunction are overwrite in the original matrix, which in this case it is called `hamiltonian`, hence the matrix is saved after the computation as well as the computed eigenvalues.

```
hamiltonian = 0 !kinetic matrix +potential matrix
do i = 1, n_points
    do j = 1, n_points
        if (i == j) then
            hamiltonian(i,j) = 2._pr *hbar* hbar/(2._pr*mass *dx**2) + v(i)
        elseif (i == j+1) then
            hamiltonian(i, j) = -1._pr*hbar* hbar/(2._pr*mass *dx**2)
        elseif(i == j-1) then
            hamiltonian(i, j) = -1._pr*hbar* hbar/(2._pr*mass *dx**2)
        end if
    end do
end do
call eigenvalues(hamiltonian, n_points, work, eigen, rwork,  lwork, info)
```

Listing 2. FORTRAN code for initializing the matrix and compute the correspondent eigenvalues and eigenfunctions

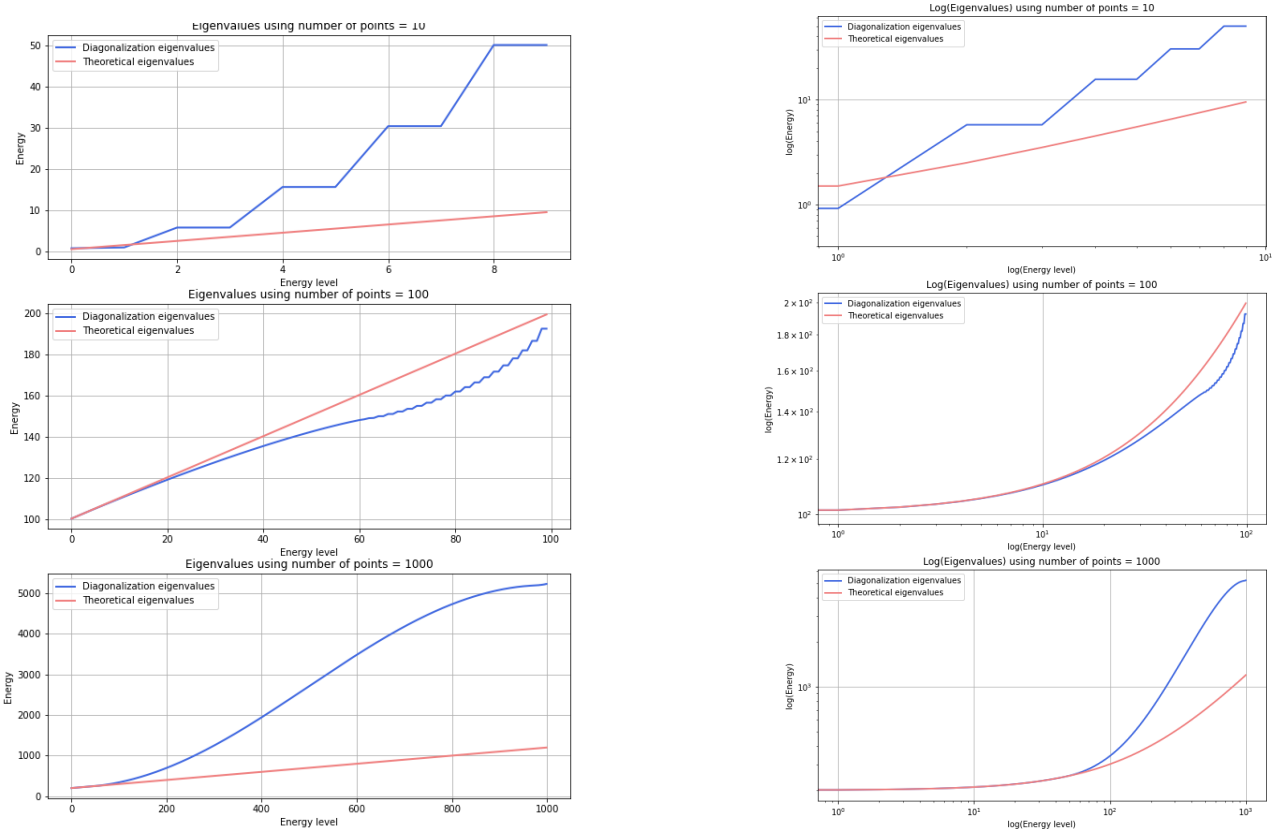
The eigenvalues are computed in FORTRAN also theoretically by knowing that the energies are given by Equation 2. At the same time, the theoretical eigenfunctions are also computed, given Equation 4, in PYHTON by noticing that the Hermite polynomials can be written in a recursive formula.

The FORTRAN main code was run using the library `subprocess` in PYTHON with $L = 10$, $\hbar = 1$, $\omega = 1$ and $n\_points = [10, 100, 1000]$. Furthemore, the CPU time were computed in a range of $n\_points = range(10, 1000, 10)$ where we effectively fit the results in GNUPLOT using $f(x) \sim N^3$.

3 Results

Figure 1 shows the eigenvalues obtain for the several $n\_points = [10, 100, 1000]$ compared with the theoretical ones. As it can be seen the discretization plays an important role in determining the correct eigenvalues. Indeed, when the number of points are low, the energies obtained are not the best one whereas if we increase the number of points this leads into a better discretization, and then better results. Nevertheless this is up to a certain threshold, between 100 and 200 points, where the eigenvalues are higher than the theoretical ones.

This divergent behaviour might be due to the fact that a finite interval is used: when lowering the spacing the system is more similar to a particle in an infinite well since we are observing the quadratic potential more and more flat. Nevertheless, we were interesing in the first eigenvalues and indeed as shown in Table 1 the first 3 eigenvalues matches with the theoretical one.



(a) Computational and theoretical Eigenvalues

(b) Computational and theoretical Eigenvalues in logscale

Figure 1. Eigenvalues

Figure 2 shows the first 7 level of the harmonic oscillator with its eigenfunctions. AS expected the number of nodes increases with increasing quantum number and energy and the ground state has energy $E = \frac{1}{2}\hbar\omega$

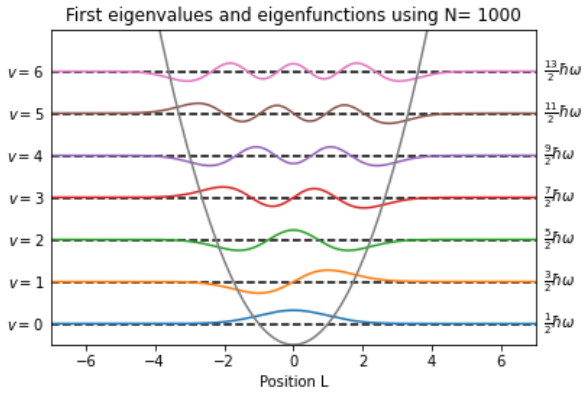
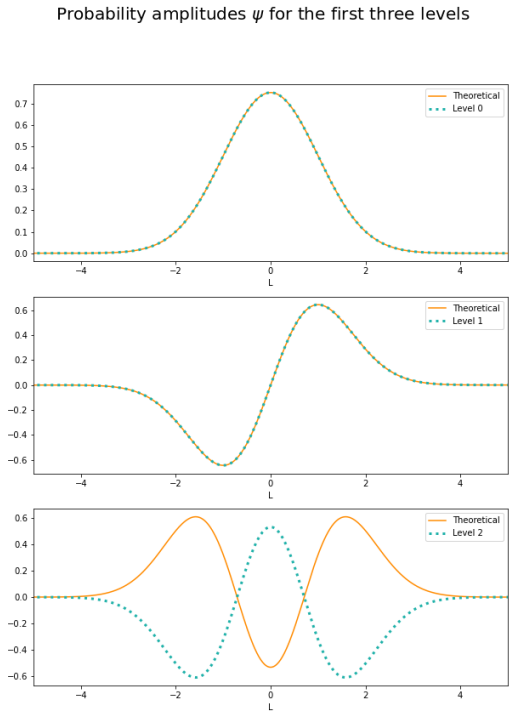


Table 1

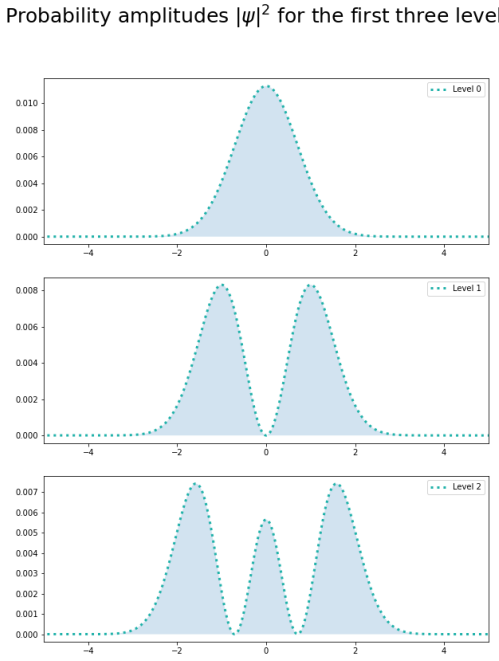
Level	Computational E_n	Theoreticall E_n
1	0.49998747464744236	0.5
2	1.4999373719868916	1.5
3	2.4998371635275367	2.5

Figure 2. Computed eigenfunctions and eigenvectors

Figure 3a shows the comparison between the first eigenfunction computed with the theoretical ones. Notice that since the eigevectors provided by `zheev` are normalized to 1 we have to multiplied each value per a factor of $1/\sqrt{dx}$. It can be seen that for the third eigenvalue the results are the opposite. However, comparing the square modulus of the wave function, Figure 3b, we can see that indeed they correspond to the same quantity and indeed this is a physical results.



(a) Computational and theoretical Eigenfunctions



(b) Computational and theoretical of the square amplitude of the Eigenfunctions

Finally, Figure 4 shows the computational time when increasing the dimension of the matrix, meaning lowering the spacing, with the correspond fitted function $f(x) \sim N^3$. It can be seen, apart from some points, that indeed this scaled as N^3 which for large matrix this becomes a really highly complexity system.

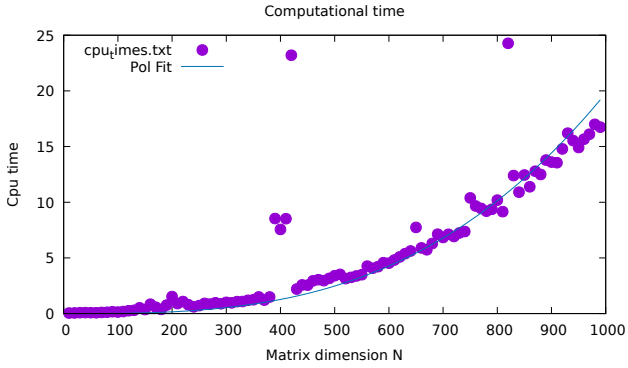


Figure 4. Cpu time

4 Conclusion

In conclusion, the first eigenvalues and eigenvectors were successfully found using LAPACK. Regarding the correctness, the program does what it is expected to do and furthermore this can be inspect by some checks during the program such as initialization of the matrix. Moreover, in order to avoid unwanted behaviour, `IMPLICIT NONE` was used and the documentation seems proper in order to understand it and reuse it, no compilation and memory issues were found. FORTRAN is already a flexible program that can be easily be implemented, and this code is quite flexible for any quantum harmonic oscillator problem. In fact the parameters involving the physical problem can be easily tuned, as well as the size L and the n_points.

The chosen discretization $n\_points = 1000$ is accurate in obtain the first eigenfuctions and eigenvalues. However, it was found that the program highly depends on the n_points: increasing them leads into better result, nonetheless it was also found that higher number of points leads into higher computational time $\mathcal{O}(N^3)$ and for larger level of energies we miss the underlying physical phenomena, turning a quantum oscillator problem into a infinite square well defined one.