Exercise 5: *Numerical computation of Wave-Packet of time dependent quantum harmonic oscillator*

Iriarte Delfina (1231682)

Quantum Information

November 24, 2020

Abstract

In this report, the time-dependent quantum harmonic oscillators is solved by transforming it correspondent Hamiltonian by a set of unitary operators. This is called as the **Split Operator Method** which allows to split the Hamiltonian into a kinetic energy part and a potential energy part regardless of not commute. In particular, this will be solved using FORTRAN and by exploiting the SUBROUTINE FFTW that provides a way to compute a **Fast Fourier Transform**.

1 Introduction

Quantum wave-packet dynamics, that is, the evolution of the spatial distribution of a quantum particle, plays an important role of the simulation of **many body systems**. Moreover, time-dependent harmonic oscillators arise in quantum mechanical systems such as optical trapping of different objects like atoms and molecules, living cells, and more. Generally, consider the time-dependent Schrodinger equation:

$$i\hbar\frac{\partial}{\partial_t}\Psi(t) = \hat{\mathcal{H}}\Psi(t) \tag{1}$$

where $\hat{\mathcal{H}}$ is the Hamiltonian for the motion of a particle interacting with an external time-dependent potential $V(t)$, namely:

$$\hat{\mathcal{H}} = \frac{\hat{P}^2}{2m} + V(t) \tag{2}$$

In our case the Hamiltonian is given by:

$$\mathcal{H} = \frac{P^2}{2m} + \frac{1}{2}m\omega^2\left(x - \frac{t}{T}\right)^2 \tag{3}$$

The formal solution of equation 1 is given by the **time evolution operator** $\hat{U}$ which allows to propagate the wave function from time t_0, $\Psi(t_0)$, at any time t:

$$\Psi(t) = \hat{U}(t, t_0)\Psi(t_0) \tag{4}$$

Since the Hamiltonian is **time-dependent** we have that:

$$\hat{U}(t, t_0) = \hat{T}\exp\left(\frac{-i}{h}\int_{t_0}^{t}\hat{\mathcal{H}}(t')dt'\right) = \hat{T}\exp\left(\frac{-i}{h}\int_{t_0}^{t}[\hat{K} + V(\hat{t}')]dt'\right) \tag{5}$$

where $\hat{T}$ is the time-ordering operator. In general, the Hamiltonian **does not commute** with itself at a different time, i.e., $[\hat{H}(t), \hat{H}(t')] \neq 0 \iff t \neq t'$. Moreover, notice that the potential is also spatial dependence $\hat{V} = V(x,t)$ hence $\hat{K}$ and $\hat{V}$ does not commute. Nevertheless, exploiting the Baker-Campell-Hausdorff relation and the Split Operator method we can approximate it and hence **split** the time evolution operator that propagates the system from t to $t + \Delta t$:

$$\Psi(r,t+dt) \sim [e^{-i\hat{V}\Delta T/2} e^{-i\hat{T}\Delta T} e^{-i\hat{V}\Delta T/2}] + O(\Delta t^3) \tag{6}$$

Then we can address each part of the solution in chunks, first in position space, then in momentum space, and finally in position space again by using the **Fourier transform**. The solution will be given by:

$$\Psi(x,t+dt) \sim [U_x(x,t)\mathcal{F}^{-1}U_p(p,t)\mathcal{F}U_x]\Psi(x,t) \tag{7}$$

where we defined:

$$\hat{U}_x(x,t) = \exp\Big(-i\frac{V(\hat{x})\Delta t}{2}\Big) \tag{8}$$

$$\hat{U}_p(p,t) = \exp\Big(-iT(p)\Delta t\Big) \tag{9}$$

With the use of cleverly **diagonalized exponential matrices** Split operator method can be calculated a lot faster than other available methods.

In this project, the **time-dependent Schrödinger equation** for the **ground state of the quantum harmonic oscillator** in $1D$ is computed by means of the Split Operator method in FORTRAN. In particular, we used the **Fast Fourier transform**, which improve the computational effort when calculating the time evolution of the wave function given by Equation 7. Finally, the evolution of the **square norm** of the wave function and the **average position** at each time step are plotted using GNUPLOT.

2 Code development

In order to compute the time evolution of the quantum harmonic oscillator we first need to found the correspondent ground state at time $t = 0$. This is done by implementing last week exercise, where the eigenvalues and eigenvectors are computed using the **Finite Difference Method** for the time-**independent** quantum harmonic oscillator. Once the wave function of the ground state is found, the time propagation is computed by implementing the **Split Method**. In particular, the parameters of the system `L`, `n_points`, `time_split` and `total_time` are read from an external file.

Notice that the space grid, of dimension `dim_space = n_points +1`, is such that `x_i = -L + (i-1)` Δ`x` for every `i` $\in$ `{1, ...dim_space}`. Therefore, the momentum grid is such that $p\_i = \frac{2\pi i}{dim\_space\Delta X}$ for `i` $\in$ `{1, ...dim_space}`. As a final remark, notice that most of FT implementations have the negative frequencies in the second half of the range. For this reason we compute the moments for `[0, (dim_space-1)/2]` and then for `[(dim_space+1)/2, dim_space]` as shown in Listing 1-

```
max_p = 2._pr*pi/dx
! Initializing vector of moments
```

```fortran
do i=1,(dim_space-1)/2
    moments(i) = max_p/(dim_space) * (float(i)-1)
enddo
do i=(dim_space+1)/2, dim_space
    moments(i) = max_p/(dim_space) * (float(i)-1) - max_p
enddo
```

Listing 1. Vector of moments in FORTRAN

A grid `grid_psi`, of size `dim_space` × `dim_time`, is built in order to record the evolution of the eigenstate for the several time steps in each columns. The first column will contain the **normalized** eigenstate of the ground state at time t = 0 previously computed. The split operator algorithm goes as follows: first the ground state is propagate by Propagator 8 and then the Fast Fourier transform is computed by using the SUBROUTINE FFTW[1]. At this stage the wave function has move from the $|x\rangle$ representation to the $|p\rangle$ one. In order to transform back, the just computed Fourier transform is propagated by using Propagator 9 and the anti-Fourier is applied. Finally, wave function at time $t + \Delta t$ is obtained by propagating the anti-Fourier by means of Propagator 8. This is done iterative for all the correspondent time as shown in Listing 2.

```fortran
  !ground state normalized
    psi_0 = hamiltonian(:,1)/sqrt(dx)
 !Create grid with first state being the ground state
    grid_psi(:,1) = psi_0
        do i = 2, dim_time
        !exp(-i*V(x)*delta_t/2)
        temp(:) = exp(cmplx(0._pr, -0.5_pr*delta_t*0.5_pr*(x(:) - i * delta_t/total_time)**2)) * grid_psi(:,i-1)

        !Fourier from x to p
        call dfftw_plan_dft_1d(plan,dim_space,temp,fft1,FFTW_FORWARD,FFTW_ESTIMATE)
        call dfftw_execute_dft(plan, temp, fft1)
        call dfftw_destroy_plan(plan)

        !exp(-i*kinetic*delta_t)
        temp2(:) = exp(cmplx(0._pr, -delta_t*(moments(:)**2)*0.5_pr)) * fft1(:)

        call dfftw_plan_dft_1d(planb, dim_space, temp2, fft2, FFTW_BACKWARD, FFTW_ESTIMATE)
        call dfftw_execute_dft(planb, temp2, fft2)
        call dfftw_destroy_plan(planb)

        !issues with normalization
        fft2(:) = fft2(:) / dim_space

        !exp(-i*V(x)*delta_t/2)
         grid_psi(:,i) = exp(cmplx(0_pr, -0.5_pr*delta_t*0.5_pr*(x(:)-i*delta_t/total_time)**2)) * fft2(:)

    enddo
```

Listing 2. Calculation of time evolution of the ground state of the Quantum Harmonic Oscillator in FORTRAN

The square norm amplitude of the wave function is computed as shown in listing 3.

```fortran
 !Square of the eigenfunctions
    do ii=1,dim_space
        grid_psi_square(ii,:) = abs(grid_psi(ii,:))**2
    enddo
```

Listing 3. Calculation of the square norm amplitude of the wave function in FORTRAN

[1] http://fftw.org/

Finally, the average position is calculated as shown in Listing 4. Moreover, all the results are saved in their correspondent file and, the square amplitude of the wave function and the mean position are plotted using GNUPLOT.

```fortran
    !lets now compute the average
    do i = 1, dim_time
        mean = 0._pr
        norm = 0._pr
        do j = 1, dim_space
            mean = mean + (-L + (float(j) - 1)*dx)*dx * abs(grid_psi(j, i))**2
            norm = norm + dx * abs(grid_psi(j,i))**2
        enddo
        mean = mean/norm
    enddo
```

Listing 4. Generation of the harmonic potential in FORTRAN

3 Results

Figure 1 shows the time evolution of the square norm of the ground state of the quantum harmonic oscillator using the parameters `L = 5`, `n_points = 1000` , `T_tot = 20`, `time_split = 100` . The probability density remains with a Gaussian shape which it is what we expected.

Nevertheless it was found that when using a high time step the system becomes unstable and the results computed are not reliable For small `T_tot` the evolution of the wave function is not appreciate, since not meaningful translation can be seen. Moreover, recall that when computing the eigenfunctions and the eigenvalues of the time-independent Schrodinger equation the space step should be properly initialized since since for small values it produce wrong results. Hence it is really important that this two parameters should be tuned properly.

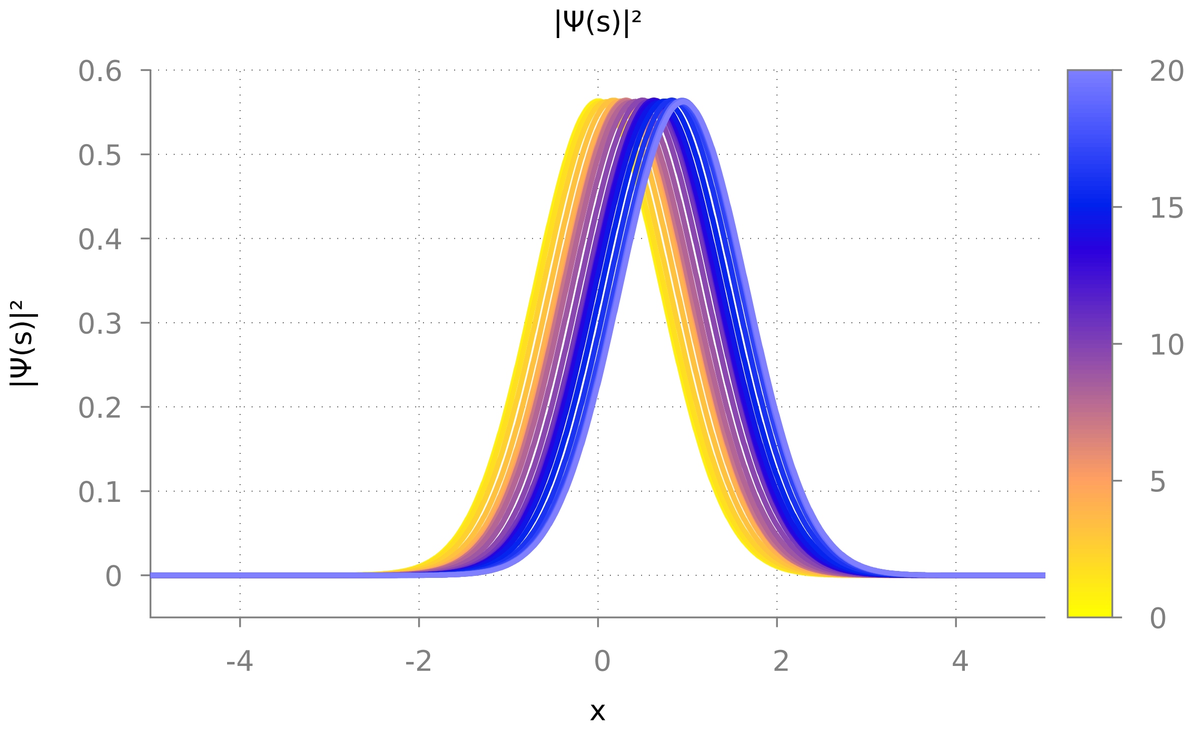
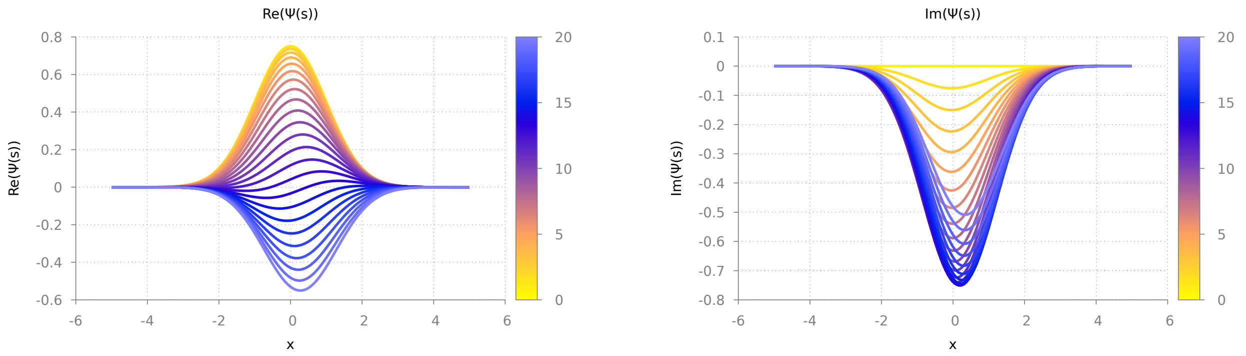


Figure 1. Wave packet evolution of the ground state for the square norm.

Figure 2 shows the time evolution of the Real component and the imaginary component of the wavefunction.



(a) Real component

(b) Imaginary component

Figure 2. Time evolution of the ground state of the quantum harmonic oscillator.

Figure 3 shows the time evolution of the average position of the particle. It can be seen that the result resembles to the classical case of the harmonic oscillator in a moving frame with constant velocity.

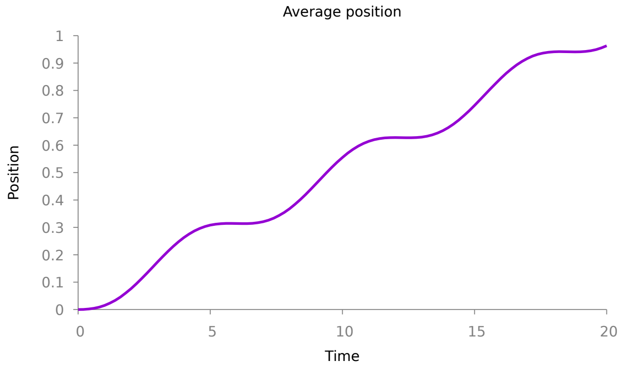


Figure 3. Mean position at each time step

4 Conclusion

In conclusion, the wave-packet of the ground state of the quantum harmonic oscillators in 1D was successfully implemented by using the Split operator method. In particular, the ground state of the Hamiltonian was found using the LAPACK library and code produce from last week exercise. It was founded that the method produce correct result when the space steps and time steps are tuned correctly. Furthermore, the CPU time is indeed really fast and there is not any issue of memory or compilation.