Time Dependent Schrödinger Equation

Assignment 5

Gabriel Amorosetti[1*]

[1]MSc Physics of Data, Università degli Studi di Padova

Abstract

This study investigates the time-dependent Schrödinger equation applied to a one-dimensional quantum harmonic oscillator, introducing a time-dependent Hamiltonian with a moving potential center. Using the split-step Fourier method, the evolution of the wave function is computed numerically, offering an efficient approach to handling the interplay between kinetic and potential energy terms. By leveraging the Trotter-Suzuki decomposition, the numerical scheme achieves higher accuracy in approximating the time-evolution operator. The results provide insights into the dynamical behavior of quantum systems under time-dependent perturbations, demonstrating the utility of Fourier-based methods in quantum mechanics.

Introduction

The quantum harmonic oscillator is the quantum analogue of the classic harmonic oscillator and is one of the few quantum systems for which an exact analytical solution is known. This fundamental problem in quantum mechanics models the behaviour of a particle in a quadratic potential well. The Hamiltonian of this particle, introduced in the previous assignment, is :

$$\hat{H} = \frac{\hat{p}^2}{2m} + \frac{1}{2}m\omega^2\hat{q}^2 \tag{1}$$

with the mass m, the angular frequency ω, the position $\hat{q}$ and the impulsion $\hat{p} = -i\hbar\nabla$. In order to study this problem, one should solve the Schrödinger equation :

$$\hat{H}\Psi(x) = E\Psi(x) \tag{2}$$

where $\hat{H}$ is the Hamiltonian operator, $\Psi(x)$ are the eigenfunctions, and E are the corresponding eigenvalues (energy levels), that are crucial to describe the quantum states of the system.

In order to study how quantum states evolve with time, we now consider the time-dependent one-dimensional quantum harmonic oscillator defined by the Hamiltonian :

$$\hat{H}(t) = \frac{\hat{p}^2}{2m} + \frac{\omega^2\left(\hat{q} - q_0(t)\right)^2}{2m} \tag{3}$$

with $q_0(t) = \frac{t}{T}$ and $t \in [0 : T]$.

This report focuses on solving the time-dependent Schrödinger equation for the quantum harmonic oscillator numerically, specifically using the split-step Fourier method, which can efficiently compute wavefunction evolution over time.

Methodology

In this context,

- $\hbar$ and m will be set as natural units, therefore equals to 1,
- the problem will be limited to the case of one particle in one spatial dimension.

In order to study the time evolution of $|\Psi(x,t)\rangle$, one needs to solve the time-dependent Schrödinger equation :

$$i\hbar\frac{\partial|\Psi(x,t)\rangle}{\partial t} = \hat{H}|\Psi(x,t)\rangle \tag{4}$$

The general solution to the time-dependent Schrödinger equation is given by :

$$|\Psi(x,t)\rangle = \hat{U}(t)|\Psi(x,0)\rangle \tag{5}$$

with $\hat{U}(t)$ the time-evolution operator :

$$\hat{U}(t) = e^{-i\hat{H}t/\hbar} \tag{6}$$

and $|\Psi(x,0)\rangle$ the wave-function at the initial time $t = 0$.

While $\hat{H}(t)$ is time-dependent, $\hat{U}(t)$ can still be written in the same form as long as it describes the evolution in a time interval Δt small enough :

$$\hat{U}(\Delta t) \sim e^{-i\hat{H}(t)\Delta t/\hbar} \tag{7}$$

Following this description, the state evolution at a time $t = N\Delta t$ is obtained by successive application of the operator $\hat{U}(\Delta t)$:

$$|\Psi(x,t)\rangle = \hat{U}(\Delta t)^N|\Psi(x,0)\rangle \tag{8}$$

Using the Baker–Campbell–Hausdorff formula, the equation 7 can be rewritten as :

$$\hat{U}(\Delta t) = e^{-i\hat{T}\Delta t/\hbar} \quad e^{-i\hat{V}(t)\Delta t/\hbar} + \mathcal{O}\left(\Delta t^2\right) \tag{9}$$

*gabriel.amorosetti@studenti.unipd.it

Due by : December 2, 2024

which can be rewritten again thanks to the Trotter-Suzuki decomposition, in order to improve the accuracy, to :

$$\hat{U}(\Delta t) = e^{-i\hat{V}(t)\Delta t/2\hbar} e^{-i\hat{T}\Delta t/\hbar} e^{-i\hat{V}(t)\Delta t/2\hbar} + \mathcal{O}\left(\Delta t^3\right) \tag{10}$$

The decomposition described in the equation 10 is useful as it splits the Hamiltonian into more manageable parts. Indeed, the kinetic term $\hat{T}$ in the Hamiltonian involves the second derivative of the wave-function with respect to position. This can be handled in momentum space via a Fourier transform, giving its momentum space representation, where the operator becomes a multiplication operation. However, the potential term $\hat{V}(t)$ is diagonal in real space, meaning it acts directly on the wave-function without needing a Fourier transform.

The wave-function in the Fourier transformed space is expressed as follows :

$$\mathscr{F}\Psi(x) = \frac{1}{2\pi\hbar} \int e^{-ip_x x} \Psi(x) dx = \tilde{\Psi}\left(p_x\right) \tag{11}$$

Therefore, with this representation, $\hat{U}(\Delta t)$ can be described by :

$$\hat{U}(\Delta t) = e^{-i\hat{V}(t)\Delta t/2\hbar} \quad \mathscr{F}^{-1} e^{-i\hat{T}\Delta t/\hbar} \quad \mathscr{F} e^{-i\hat{V}(t)\Delta t/2\hbar} \tag{12}$$

The representation of $\hat{U}(\Delta t)$ as expressed in the equation 12 will be used when implementing the Split Step method in order to compute the time evolution of $|\Psi(x,t)\rangle$, which will be detailed in the following section.

Implementation

In order to compute the time evolution of the wave-function $|\Psi(x,t)\rangle$ describing the time-dependent one-dimensional quantum harmonic oscillator defined by the Hamiltonian :

$$\hat{H}(t) = \hat{T} + \hat{V}(t) = \frac{\hat{p}^2}{2m} + \frac{m\omega^2 \left(\hat{x} - x_0(t)\right)^2}{2} \tag{13}$$

we develop a Python code implementing the Split Step method. The idea of the Split-Step method is that time evolution can be approximated by alternating between two steps. First, evaluating the wave function using the kinetic operator in the Fourier space, and then evaluating it using the potential operator in real space.

The input parameters are :

- `x_min, x_max` : the minimal and maximal values of the space interval,
- `Nslice` : the number of slices in space,
- `Ntslice` : the number of slices in time,
- `T` : the timescale for $q_0(t) = \frac{t}{T}$,
- `t_sim` : the duration of the simulation, `t_sim` $\in [0 : T]$.

From these parameters can be derived :

- `dx = (x_max - x_min) / Nslice` : the spatial step,
- `dt = t_sim / Ntslice` : the time step,
- `x = np.linspace(x_min, x_max, Nslice, endpoint = False)` : the grid of spatial points,
- `p = (2 * np.pi / (x_max - x_min)) * np.fft.fftfreq(Nslice, d=dx)` : the grid of the discretized momentum points in the momentum space (*i.e.* a grid of frequencies).

With that, some useful functions used later in the Split Step method can be defined :

```python
# Potential q0(t)
def q0(t) :
    return t / T

# Potential energy operator
def V(x, t) :
    return 0.5 * m * omega**2 * (x - q0(t))**2

# Ground state of the harmonic oscillator (n=0)
def psi0(x) :
    return (m * omega / (np.pi * hbar))**0.25 *
           np.exp(-m * omega * x**2 / (2 * hbar))
```

Figure 1: *Definition of functions*

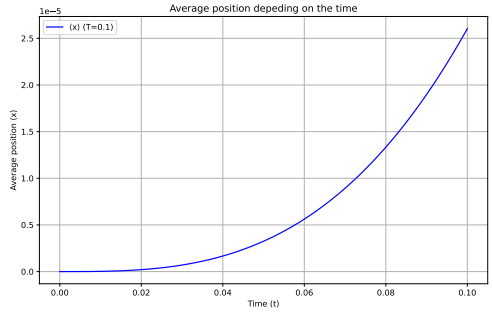
These functions return the time-dependent part of the potential term, $q_0(t) = \frac{t}{T}$, the potential operator $\hat{V}(x,t)$, and the wave function at the initial time $t = 0$, $|\Psi(x,0)\rangle$.

To solve the problem,

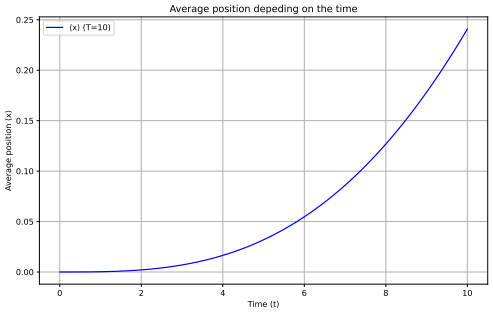
- first, the wave function is initialized : `psi = psi0(x)`,
- then, it is normalized : `psi /= np.linalg.norm(psi) * np.sqrt(dx)`,
- and finally, the time evolution is computed with the function : `evolution = split_step(psi, x, p, dt, Ntslice)`.

`split_step(psi, x, p, dt, Ntslice)` return the `evolution` and proceeds as follows : it creates a 2D array of size (`Nslice, Ntslice + 1`) and stores in its first column the wave function at the initial time `psi`, and for each time slice `t = n * dt` (`n` from 0 to `Ntslice`),

- the potential operator `Vx = V(x, t)` is applied to the wave function : `psi = np.exp(-1j * Vx * dt / (2 * hbar)) * psi` (first half step),
- the Fourier tranform is applied to the wave function : `psi_p = np.fft.fft(psi)`,
- the kinetic operator `T_p = p**2 / (2 * m)` is applied to the wave function `psi_p` : `psi_p = np.exp(-1j * T_p * dt / hbar) * psi_p`,
- the reverse Fourier transform is applied : `psi = np.fft.ifft(psi_p)`,
- the potential operator `Vx = V(x, t + dt)` is applied : `psi = np.exp(-1j * Vx * dt / (2 * hbar)) * psi` (second half step),
- the result is stored : `evolution[:, n + 1] = psi`.



(a) $T = 0.1$



(b) $T = 10$

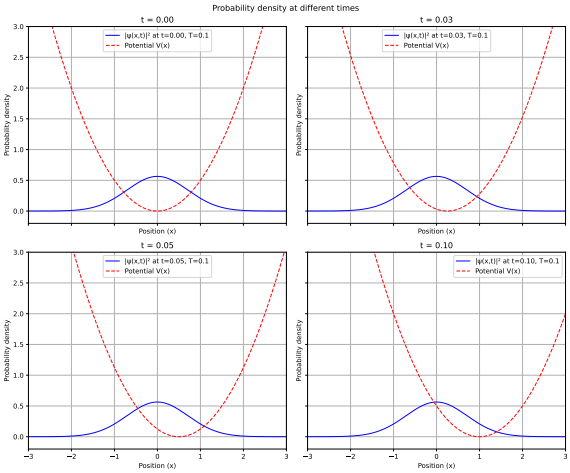
Figure 2: *Average position depending on the time*

Results

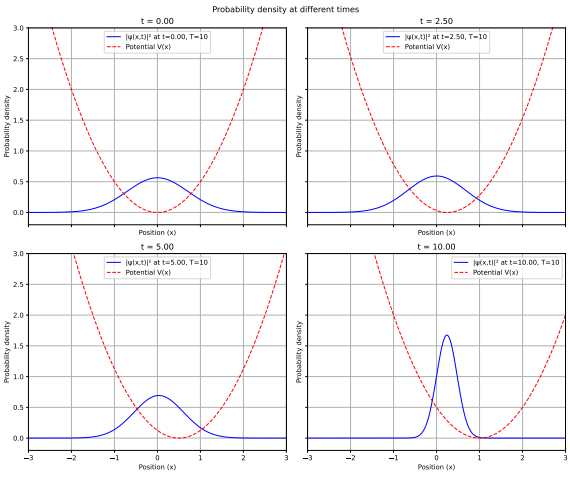
The probability density (at different times `t` $\in [0 : T]$) and the average position are evaluated for different values of `T`, and with the following parameters :

- `x_min, x_max = -4, 4`
- `Nslice = 1000`
- `Ntslice = 1000`

For low-order values of `T`, the wave function seems to be stuck in the same position or does not evolve following the potential's evolution up to $x = 1$, as shown in the figure 3 for $T \leq 10$.



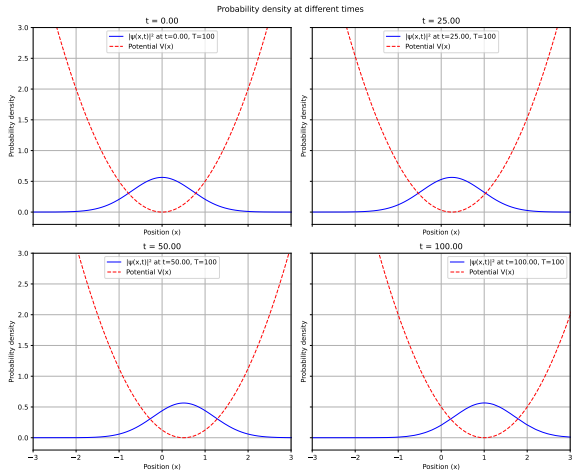
(a) $T = 0.1$



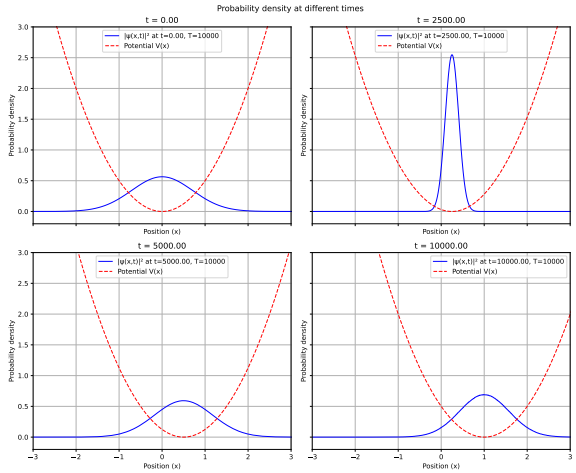
(b) $T = 10$

Figure 3: *Time evolution of the probability density*

This behaviour can be observed by looking at the average position of x depending on the time. For these values of `T`, the wave function does not indeed follow the potential up to $x = 1$.



(a) $T = 100$



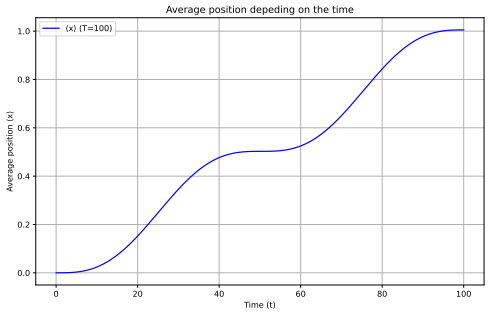
(b) $T = 10000$

Figure 4: *Time evolution of the probability density*

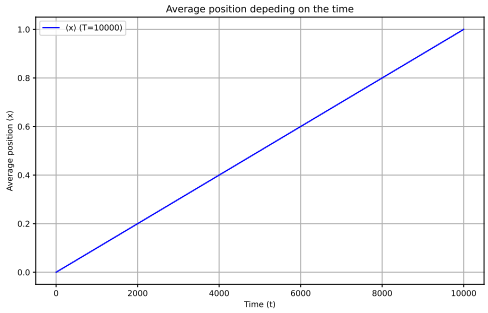
For increased values of `T`, from 100 to 10000, the wave function reaches $x = 1$ as observed in the figure 4.

One observation from the figure 5 is that for higher values of `T`, the shift of the wave function from $x = 0$ to $x = 1$ tends to become a linear shift depending on the time.

Finally, if `T` is increased again, above 15900, the evolution of the wave function becomes more and more chaotic, as observed in the figures 6 and 7.



(a) $T = 100$



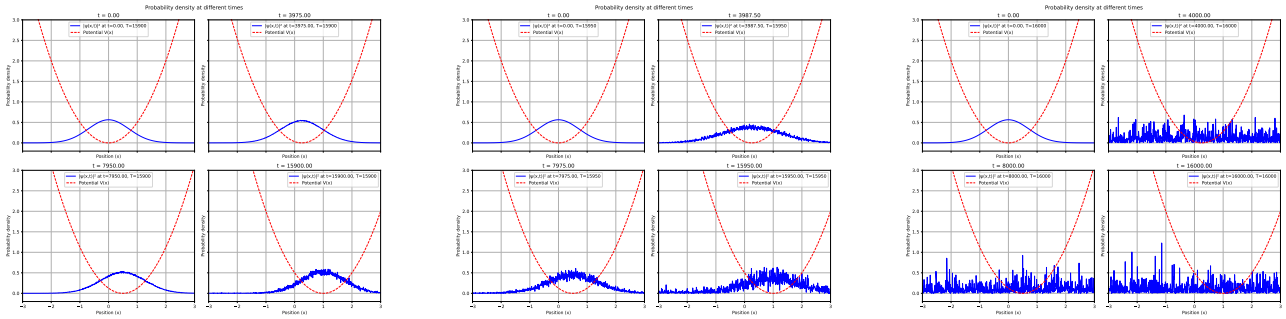
(b) $T = 10000$

Figure 5: *Average position depending on the time*

Conclusion

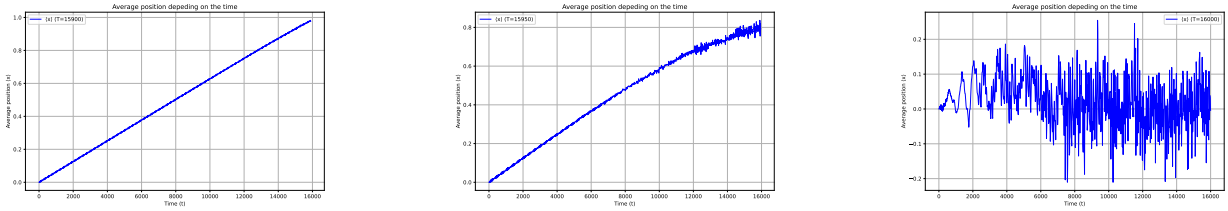
This study shows the different behaviours of the wave function evolution for different values of `T`. As defined in the code, `t_sim = T` and therefore $\Delta t = \frac{\texttt{T}}{\texttt{Ntslice}}$

- If $\Delta t << 1$, the time step is too small and $q_o(t)$ does not allow evolution of the wave function,
- if $\Delta t \approx 1$, the wave function evolves linearly following the time dependant potential,
- if $\Delta t >> 1$, the evolution of the wave function becomes chaotic as the precision of the decomposition of Δt is limited to $\mathcal{O}(\Delta t^3)$ (equation 10), showing the limits of this numerical implementation.



(a) $T = 15900$ **(b)** $T = 15950$ **(c)** $T = 16000$

Figure 6: *Time evolution of the probability density*



(a) $T = 15900$ **(b)** $T = 15950$ **(c)** $T = 16000$

Figure 7: *Average position depending on the time*