Quantum Dynamics - Homework

Albert Makhmudov

19/03/2025

Data Availability

The source code of this project as well as the example input files are available at the corresponding Github page. There one could also find the example output files and the detailed instructions on how to run and compile the code. The example output files are `gif` animations of the wavepacket propagation that complement the results in Figure 1. The source code of this project is written in `Fortran90`, whilst the figures were generated using `gnuplot`.

Code Overview

Input

In order to run the code, one should provide two input files, namely `wavepacket` and `potential`. It's important to name the files this way.

The `wavepacket` file should contain the following parameters: number of lattice points, initial position of the wavepacket, the initial width, propagation time step, total number of steps, snapshot frequency, total number of coefficients and the coefficients themselves.

The `potential` file should specify the type of a potential, e.g. harmonic and double well are supported, as well as the angular frequency in case of the harmonic potential and the barrier height in case of the double well potential. In both input files, each parameter should be written on a separate line. The example input files can be found in `harmonic/wavepacket` and `harmonic/potential`, respectively.

Wavepacket Initialisation and Normalisation

In this project, instead of using a gaussian wavepacket, the wavepacket is initialised as a superposition of the eigenstates of the harmonic oscillator. At $t = 0$, the wavepacket is given by the following expression:

$$\Psi(x, t=0) = \sum_{n=0}^{\infty} c_n \Phi_n(x) \tag{1}$$

Where the eigenstates of the quantum harmonic oscillator are calculated according to the following formula:

$$\Phi_n(x) = \frac{1}{\sqrt{2^n n!}} \left(\frac{m\omega}{\pi}\right)^{1/4} e^{-m\omega x^2/2} H_n(x\sqrt{m\omega}) \quad (n = 0, 1, 2, \cdots) \tag{2}$$

H_n is the Hermite polynomial of order n. We'll touch upon the implementation of the Hermite polynomials in the next section. While initialising the wavepacket according to the Equation 1 and Equation 2 as shown in Listing 1, the calculation of a norm of the wavepacket is performed alongside. The code snippets for the wavepacket initialisation are provided below.

```
norm = 0.0d0
nmax = size(coeff) - 1
do i = -npoints/2 + 1, npoints/2
    x = dble(i) * dx
    if (i > 0) then
        j = i
    else
        j = i + npoints
    endif
    psi0(j) = 0.0d0
    do n = 0, nmax
        call factorial(n, fact)
        psi0(j) = psi0(j) + &
                coeff(n+1) * &
                (1.0d0 / sqrt(2.0d0**n * fact)) * &
                (mass * angfreq / pi)**0.25d0 * &
                exp(-mass * angfreq * alpha * (x-x0)**2.0d0 / 2.0d0) * &
                hermite(n, (x-x0) * sqrt(mass * angfreq))
    end do
    norm = norm + abs(psi0(j))**2.0d0 * dx
end do
```

Listing 1: Initialisation of the wavepacket in the `initpsi` subroutine.

As soon as the norm of the wavepacket is calculated, the wavepacket is normalised as shown in Listing 2. The normalisation is performed by dividing each element of the wavepacket by the square root of the norm. This ensures that the wavepacket is properly normalised.

```
norm = 1.0d0 / sqrt(norm)
do i = -npoints/2 + 1, npoints/2
    x = dble(i) * dx
    if (i > 0) then
        j = i
    else
        j = i + npoints
    endif
    psi0(j) = norm * psi0(j)
end do
```

Listing 2: Normalisation of the wavepacket in the `initpsi` subroutine.

Due to the need of computing the factorials, the corresponding subroutine is implemented as shown in Listing 3. The factorial is calculated up to the nth order.

```
    result = 1.0d0
    do i = 1, n
       result = result * i
    end do
```

Listing 3: Calculation of the factorial up to the nth order in the `factorial` subroutine.

Hermite Polynomials

The Hermite polynomials are calculated using the following recurrence relation:

$$H_n(y) = 2yH_{n-1}(y) - 2(n-1)H_{n-2}(y) \quad (H_0(y) = 1,\ H_1(y) = 2y) \tag{3}$$

Where H_0 isn't calculated explicitly. The reason to use Hermite polynomials in the wavepacket initialisation is due to the fact that the eigenstates of the quantum harmonic oscillator are expressed in terms of the Hermite polynomials as shown in Equation 2. Specifically, the physicist's Hermite polynomials are used in this project. The code snippet for the calculation of the Hermite polynomials is provided in Listing 4.

```
if (n == 0) then
Hn = 1.0d0
elseif (n == 1) then
Hn = 2.0d0 * y
else
H0 = 1.0d0
H1 = 2.0d0 * y
do i = 2, n
   H_prev = H1
   H1 = 2.0d0 * y * H1 - 2.0d0 * (i - 1) * H0
   H0 = H_prev
end do
Hn = H1
endif
```

Listing 4: Calculation of the Hermite polynomials up to the nth order in the `hermite` function.

Results and Discussion

The wavepackets were propagated a the bottom of the harmonic potential with the angular frequency $\omega = 0.2825$ fs^{-1}. More over they were propagated for 5000 steps with a 0.1 fs time step. The initial width of the wave packet was equal to 1.0 Bohr^{-2} and the snapshots were taken each 10th step. The wavepackets were propagated with different number of eigenstates, namely 1, 2, 3, and 5. The time evolution of the wavepackets is shown in Figure 1.

The wavepackets with a single eigenstate are stationary, whilst the wavepackets with multiple eigenstates are oscillating. The oscillations are due to the fact that the wavepacket is a superposition of the eigenstates of the harmonic oscillator. The more eigenstates are included in the wavepacket, the more oscillations are observed. The wavepacket with 5 eigenstates oscillates the most. The oscillations are periodic and the period of the probability density is calculated below.

The period of the probability density for the harmonic oscillator case can be calculated from the angular frequency as follows:

$$T = \frac{2\pi}{\omega} \tag{4}$$

Therefore the periods T of the corresponding probability densities are equal to 22.23 fs.

Speaking of the symmetry, the wavepackets with the even number of eigenstates have symmetric probability densities of the wavefunction. On the other hand, the odd number of eigenstates results in the probability densities being assymetric as can be seen in Figure 1.

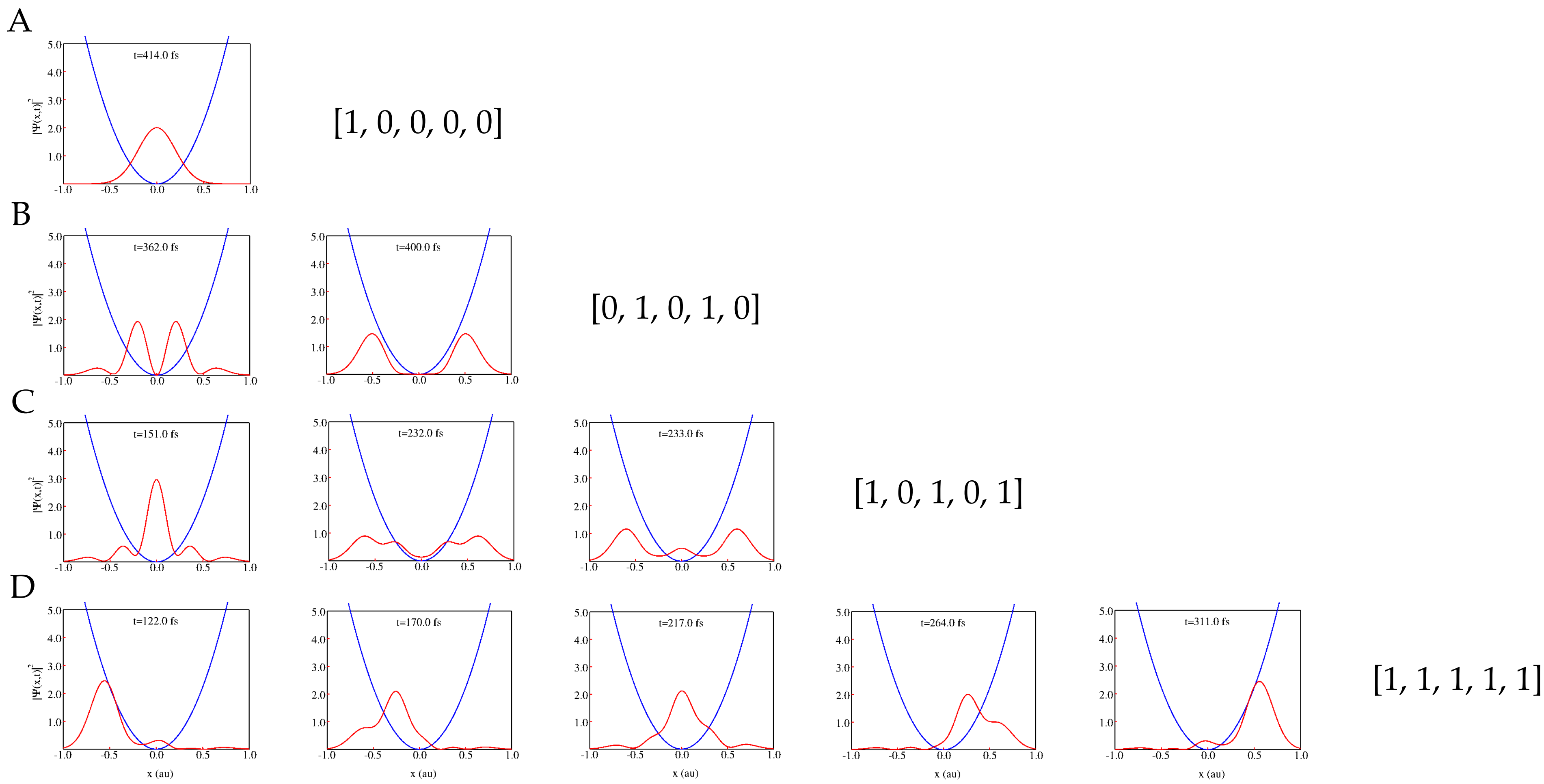

Figure 1: Time evolution of the wavepackets with different number of eigenstates. The wavepackets were propagated using the **(A)** $[1, 0, 0, 0, 0]$, **(B)** $[0, 1, 0, 1, 0]$, **(C)** $[1, 0, 1, 0, 1]$, and **(D)** $[1, 1, 1, 1, 1]$ coefficients.

The underlying reason for this is the wavefunction itself. The soutions alternate between even and odd functions around the $x = 0$ resulting in different symmetries of the probability densities [1].

While the particle with one eigenstate spends the majority of time at the bottom of the potential, the particles with multiple eigenstates oscillate. The chance of finding the particle outside of the potential increases as well. For instance, for a particle with one eigenstate this probability is around 16 % [2].

Generative AI Usage

In this work, ChatGPT large language model was used to check grammar and spelling of the main body of text, whilst Perplexity was utilised for the sake of literature search.

Acknowledgments

This project is based on data and instructions provided by Arjan Berger available at the aforementioned Github page.

References

[1] R. Nave. Quantum harmonic oscillator: Wavefunctions, 2025-03-17. URL http://hyperphysics.phy-astr.gsu.edu/hbase/quantum/hosc5.html. Georgia State University.

[2] Libretexts. The quantum harmonic oscillator, 2025-03-17. URL https://phys.libretexts.org/Bookshelves/University_Physics/University_Physics_(OpenStax)/University_Physics_III_-_Optics_and_Modern_Physics_(OpenStax)/07%3A_Quantum_Mechanics/7.06%3A_The_Quantum_Harmonic_Oscillator. University Physics III - Optics and Modern Physics (OpenStax).