Modeling a quantum harmonic oscillator using Julia

Tatyana R. Velieva[a,b], Arseny V. Fedorov[b]

[a]Plekhanov Russian University of Economics, 36, Stremyanny lane, Moscow, 117997, Russia

[b]Department of Applied Probability and Informatics, Peoples' Friendship University of Russia (RUDN University), 6, Miklukho-Maklaya St., Moscow, 117198, Russia

Abstract

In this paper we verified the applicability and usability of Julia programming language in the field of numerical simulation. To research the potential of Julia, the simplest problem of mathematical modeling was considered—the task of a quantum harmonic oscillator, for the solution of which the powerful library DifferentialEquations.jl was used. This library allows to quickly and conveniently find numerical solutions of various differential equations. During our research graphs and solutions corresponding to theoretical values were obtained.

As a result of the work, the applicability, capabilities of the Julia programming language and the convenience of its use in solving mathematical modeling problems were shown.

Keywords

Mathematical modeling, ODE, quantum harmonic oscillator, Schrödinger equation, Julia programming language

1. Introduction

An oscillator is a system of bodies or particles that make periodic oscillations around a stable equilibrium position.

The movement of microparticles is an important area of research in modern physics. One of the classical model problems in this area is the problem of the motion of a harmonic oscillator — a system capable of performing harmonic oscillations.

The real beginning of quantum theory comes from Max Karl Ernst Ludwig Planck (1858-1947, Göttingen, Germany), a German theoretical physicist, the founder of quantum physics, who in 1900 received a formula for correctly describing the spectral distribution of thermal radiation. Having come to the conclusion that he was unable to obtain the required formula for the distribution of radiation, Max Planck made the assumption that the harmonic oscillators considered as emitters should have energies that are not distributed as continuous variables, but that take discrete and regularly located values. Oscillators with a frequency of ω should have energy values that are multiple, i.e. n times multiplied (where $n = 0, 1, 2, 3, \ldots$) by an object called it a quantum energy denoted by $h\omega$.

Workshop on information technology and scientific computing in the framework of the X International Conference Information and Telecommunication Technologies and Mathematical Modeling of High-Tech Systems (ITTMM-2020), Moscow, Russian, April 13-17, 2020

velieva-tr@rudn.ru (T. R. Velieva); 1032193055@pfur.ru (A. V. Fedorov)

0000-0003-4466-8531 (T. R. Velieva); 0000-0002-3036-0117 (A. V. Fedorov)

1.1. Article structure

The following structure is proposed for this article: the section 1 describes general facts and information about quantum physics; the section 2 introduces the notation that will be used throughout the article and the agreement, following which the authors get the results; the section 2 contains graphs obtained by numerically solving the problem of modeling a quantum oscillator in a well; the section 2 describes the results; the section 3 summarizes the work done in the course of scientific research.

2. Harmonic oscillator

The harmonic oscillator model plays an important role, especially when studying small oscillations of systems near the position of stable equilibrium. An example of such vibrations in quantum mechanics is the vibrations of atoms in solids, molecules, etc. Consider a one-dimensional harmonic oscillator oscillating along the x axis under the action of a returning quasielastic force denoted by $F = -kx$. The potential energy of such an oscillator has the form:

$$U(x) = \frac{kx^2}{2} = \frac{m\omega^2 x^2}{2}, \tag{1}$$

where ω is the eigenfrequency of the classical harmonic oscillator. Thus, the quantum-mechanical problem of a harmonic oscillator reduces to the problem of the motion of a particle in a parabolic potential well.

The total energy of the oscillator (2) E is the sum of kinetic and potential energies:

$$E = \frac{p^2}{2m} + \frac{kx^2}{2}. \tag{2}$$

In classical physics, the frequency of oscillations of a harmonic oscillator, where m is the mass of the oscillator and k is a certain constant (for example, spring stiffness), which determines the scale of the force returning (to the equilibrium position) $F = -kx$ (x is deviation from the equilibrium position). The energy of a classical oscillator is proportional to the square of the amplitude of its oscillations and can change continuously [1].

Considering a quantum oscillator, there are a number of differences between it and an ordinary oscillator. It's description is assumed using the Hamiltonian.

The energy levels of the harmonic oscillator (4) and the wave functions (5) are determined from the Schrödinger equation (3) [2]:

$$\frac{h^2}{2m}\frac{d^2\psi_n(x,t)}{dx^2} + \frac{m\omega^2 x^2}{2}\psi_n(x,t) = i\hbar\frac{d\psi_n}{dt}, \tag{3}$$

$$E_n = \hbar\omega(n + 1/2), \quad n = 0, 1, 2, \ldots \tag{4}$$

Quantum oscillations realized in molecules, atoms, and nuclei can occur only with a fixed set of individual energies, i.e. the level spectrum of such an oscillator is discrete:

$$\psi_n(x,t) = \left(X_0 \cdot 2^n n!\sqrt{\pi}\right)^{-1/2} \exp\left(-\frac{x^2}{X_0^2}\right) \times H_n\left(\frac{x}{X_0}\right) \exp\left(-\frac{iE_n t}{\hbar}\right), \tag{5}$$

where the amplitude of the zero-point oscillations is given by (6):

$$X_0 = \sqrt{\frac{\hbar}{m\omega}}, \tag{6}$$

and Hermite polynomials in one of their representations are denoted as $H_n(x)$ (7):

$$H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n}\left(e^{-x^2}\right). \tag{7}$$

The energy levels of the quantum harmonic oscillator are equidistant and are given by the expression (4) $E_n = \hbar\omega(n + 1/2)$, with $n = 0, 1, 2, ...$, $\hbar = \frac{h}{2\pi}$ (h is the Planck constant), i.e. are located at the same energy distance from each other equal to E_n. The parameter n is also known as the number of phonons. The lowest energy of quantum oscillations (the energy of its zero-point vibrations) is $E_0 = \frac{\hbar\omega}{2} > 0$.

Thus, the only way to stop a quantum oscillator — is to eliminate it. Variable ω, setting the fundamental tone of a quantum oscillator is related to its potential energy by the classical relation $\hbar\omega^2\frac{x^2}{2} = \frac{kx^2}{2}$. Under the influence of an external perturbation, a quantum oscillator can go from one level to another. In this case, the minimum energy of absorbed and emitted quanta (energy of one phonon) is equal to $\hbar\omega$.

Applying the Heisenberg uncertainty relation, we set as an estimate of the momentum p: $p \approx \frac{\hbar}{x}$.

$$E = \frac{\hbar^2}{2mx^2} + \frac{kx^2}{2}. \tag{8}$$

For large values of x, the potential energy exceeds kinetic, while, for small values of x, kinetic energy has an inverse relation to potential. For the ground state, where the energy is minimal, we find the minimum of the function (8). The value of the variable x_{min} corresponding to the minimum is:

$$x_{min}^2 = \frac{1}{2\pi} = \frac{h}{\sqrt{km}}, \tag{9}$$

and the corresponding energy value E is of the order:

$$x_{min}^2 = \frac{h}{2\pi} = \frac{k}{km} = h\omega. \tag{10}$$

In quantum mechanics, to solve the harmonic oscillator problem, one needs to solve the Schrödinger equation (11) with potential energy (1), which has the form:

$$\Delta\psi + \frac{2m}{h^2}(E - U)\psi = 0, \tag{11}$$

Further equation (11) will be reduced to the following form:

$$\frac{dx^2}{h^2} + \frac{2m}{h^2}\left(E - \frac{m\omega x^2}{2}\right)\psi = 0, \quad -\infty < x < +\infty. \tag{12}$$

Lets define ε:

$$\varepsilon = \frac{X}{X_0}.$$

Then, the parameter necessary for further calculations ν:

$$\nu = \frac{2E}{\hbar\omega}.$$

We bring the equation (12) to the following view:

$$\frac{d^2\psi}{d\varepsilon^2} + (\nu - \varepsilon^2)\psi = 0. \tag{13}$$

The analysis shows that the wave functions (the solution of the equation (13)) are continuous and finite not for all values of the parameter ν, but only for:

$$\nu = 2n + 1, \quad n = 0, 1, 2, \dots .$$

We proceed to the analysis of the wave functions of a harmonic oscillator. As shown in the theory of differential equations with variable coefficients, wave functions, that are solutions of the equation (11), have the form:

$$\psi_n(\varepsilon) = e^{\frac{\varepsilon^2}{2}} H_n(\varepsilon), \quad n = 0, 1, 2, \dots , \tag{14}$$

where $H_n(\varepsilon)$ is the Chebyshev-Hermite polynomial of order n, described by the expression:

$$H_n(\varepsilon) = \frac{-1^n}{\sqrt{2^n n! \sqrt{\pi}}} e^{\varepsilon^2} \frac{d^n e^{-\varepsilon^2}}{d\varepsilon^2}.$$

The wave functions (14) are orthonormalized, that is, satisfy the condition:

$$\int_{-x}^{+x} \psi_n(x)\psi_m(x)dx = \delta_{mn},$$

where δ_{mn} is the Kronecker symbol:

$$\delta_{mn} = \begin{cases} 1, & m = n, \\ 0, & m \neq n. \end{cases}$$

Next, we present the form of the wave functions 15 for the first three energy levels of the quantum harmonic oscillator:

$$\begin{aligned} n = 0, \quad \psi_0(x) &= \frac{1}{\sqrt{x_0\sqrt{\pi}}} \exp\left(-\frac{x^2}{2x_0^2}\right), \\ n = 1, \quad \psi_1(x) &= \frac{1}{\sqrt{2x_0\sqrt{\pi}}} \frac{2x}{x_0} \exp\left(-\frac{x^2}{2x_0^2}\right), \\ n = 2, \quad \psi_2(x) &= \frac{1}{\sqrt{8x_0\sqrt{\pi}}} \left(\frac{4x^2}{x_0^2} - 2\right) \exp\left(-\frac{x^2}{2x_0^2}\right). \end{aligned} \tag{15}$$

To obtain the solution to the described problem, the program code was developed in the Julia programming language [3, 4, 5, 6], using the `DifferentialEquations.jl` [7] library.

Before working with Julia you must install the language, following the recommendations from the official website [8].

Then, using the `]` command in Julia's environment, the mode of working with packages is called `pkg`, after entering it you need to install the libraries used in the presented code: `DifferentialEquations` and `Plots`. This can be done using the command `install <library_name>`, where `library_name` is the name of the library to be installed.

```
using DifferentialEquations
using Plots
```

We set the necessary parameters. Julia is allowed to use Unicode characters, so in the code below we will denote variables using them:

```
const ħ = 1.0
const m = 1.0
const omega = 0.5
```

We define a function that describes Chebyshev–Hermite polynomials, where n has a value from 0 to 5:

```
function H(x, n)
n == 0 && return 1
n == 1 && return 2x
n == 2 && return 4x^2 - 2
n == 3 && return 8x^3 - 12x
n == 4 && return 16x^4 - 48x^2 + 12
n == 5 && return 32x^5 - 160x^3 + 120x
end
```

Define the function $E(n)$, Julia allows you to define functions in mathematical form:

```
E(n) = ħ * omega * (n + 0.5)
```

The function ψ follows from the analytical solution, this is done in order to correctly derive the initial conditions; if this step is not performed, the right solution for our problem will not be obtained:

```
function psi(x, n)
res  = 1 / sqrt(2^n * factorial(n))
res *= (m * omega / (pi * ħ)) |> sqrt |> sqrt
res *= exp(- m * omega * x * x / 2 / ħ)
res *= H(sqrt(m * omega / ħ)*x, n)
end
```

To get the result from Julia solvers, it is necessary to specify the function that describes our system of equations, therefore we define the function `Func!`. An exclamation mark in the name of the function indicates that the function does not change the input parameters during operation:

```
function Func!(du, u, n, x)
du[1] = (2m/ħ^2) * (0.5m * omega^2 * x^2 - E(n)) * u[2]
du[2] = u[1]
end
```

Set the time interval from −5 to 5 using the tspan vector. We determine the current value of n and declare the initial condition $u0$:

```
tspan = (-5.0, 5.0)
n = 0
u_0 = [0, psi(tspan[1], 0)]
```

In the `ode` variable we write the solution that the `ODEProblem` function returns. The input to this function is: `Func!` function describing the ODE; `u0` are the initial conditions; `tspan` is time period on which you need to find a solution; `n` is the parameter of the Chebyshev–Hermite polynomial; and the `saveat` parameter is a step in the time interval:

```
ode = ODEProblem(Func!, u0, tspan, n, saveat=0.1)
```

In the `sol` variable we write the solution that the `solve` function returns. The `ode` parameter is input to this function. Next, in `psi` write the result of `hcat` for the `u` array of the `sol` structure. In `p0` we write the prepared graph for the mode of the wave function $n = 0$:

```
sol = solve(ode)
psi = hcat(sol.u...)[2,:]
plot(sol.t, psi, label = "psi_0(x)")
```

For $n = 1$, we perform the same algorithm to obtain the graph:

```
n = 1
u_01 = [0, psi(tspan[1], 1)]
ode1 = ODEProblem(Func!, u_01, tspan, n, saveat=0.1)
sol1 = solve(ode1)
psi1 = hcat(sol1.u...)[2,:]
```

To build two graphs in one figure, we use the plot command, in the parameters we set $p0$, $p1$ are two graphs of wave functions for $n = 0$ and $n = 1$, in the layout parameter we set the dimension 1 by 2:

```
plot(sol1.t, psi1, label = "psi_1(x)")
savefig("n1_.pdf")
```

To obtain solutions and graphs for subsequent wave functions, the similar algorithm is used:

```
n = 2
u_02 = [0, psi(tspan[1], 2)]
ode2 = ODEProblem(Func!, u_02, tspan, n, saveat=0.1)
sol2 = solve(ode2)
psi2 = hcat(sol2.u...)[2,:]
plot(sol2.t, psi2, label = "psi_2(x)")
savefig("n2_.pdf")

n = 3
u_03 = [0, psi(tspan[1], 3)]
ode3 = ODEProblem(Func!, u_03, tspan, n, saveat=0.1)
sol3 = solve(ode3)
psi3 = hcat(sol3.u...)[2,:]
plot(sol3.t, psi3, label = "psi_3(x)")
savefig("n3_.pdf")

n = 4
u_04 = [0, psi(tspan[1], 4)]
ode4 = ODEProblem(Func!, u_04, tspan, n, saveat=0.1)
sol4 = solve(ode4)
psi4 = hcat(sol4.u...)[2,:]
plot(sol4.t, psi4, label = "psi_4(x)")
savefig("n4_.pdf")

n = 5
u_05 = [0, psi(tspan[1], 5)]
ode5 = ODEProblem(Func!, u_05, tspan, n, saveat=0.1)
sol5 = solve(ode5)
psi5 = hcat(sol5.u...)[2,:]
plot(sol5.t, psi5, label = "psi_5(x)")
savefig("n5_.pdf")
```

The article showed the applicability, capabilities of the programming language Julia and the convenience of its use in solving mathematical modeling problems.

The authors demonstrated the technique of researching the model of a quantum harmonic oscillator, and graphs of wave functions were obtained. The capabilities of the Julia programming language for modeling the described systems were also explorated. A numerical simulation of the process of finding a particle of a quantum harmonic oscillator is carried out, graphs are obtained that demonstrate the behavior of the wave functions of the system.

As the results of the work, we obtained graphs of wave functions that describe the corresponding vibration modes for the quantum number n from 0 to 5. The obtained graphs are presented in Figures 1, 2, 3, 4, 5, 6.

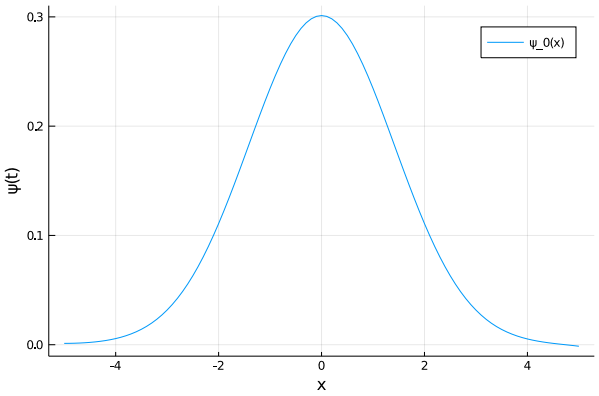

Figure 1: Graph of the wave function for the value of the quantum number $n = 0$

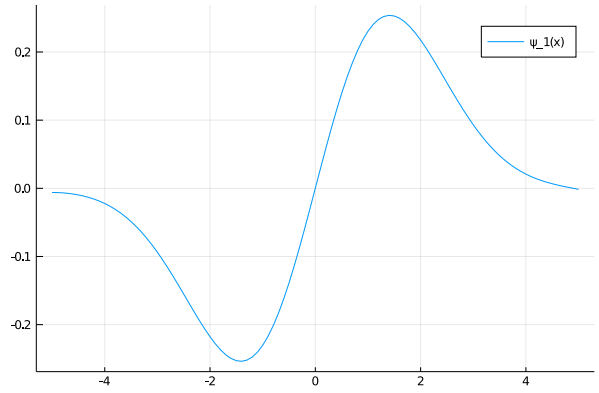

Figure 2: Graph of the wave function for the value of the quantum number $n = 1$

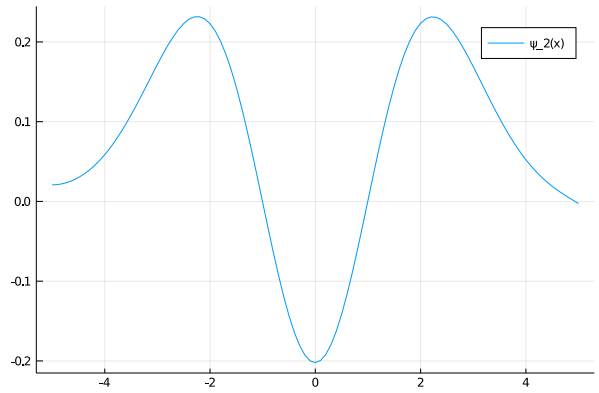

Figure 3: Graph of the wave function for the value of the quantum number $n = 2$

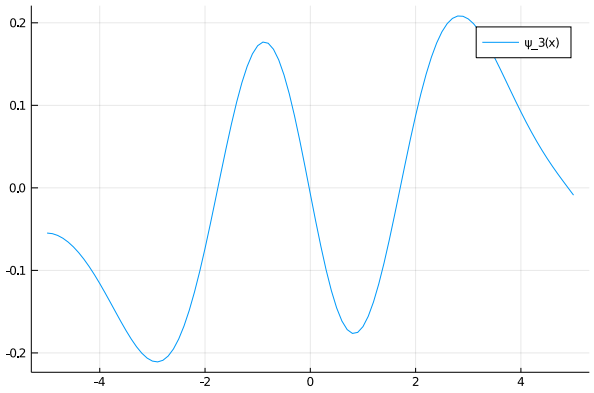

Figure 4: Graph of the wave function for the value of the quantum number $n = 3$

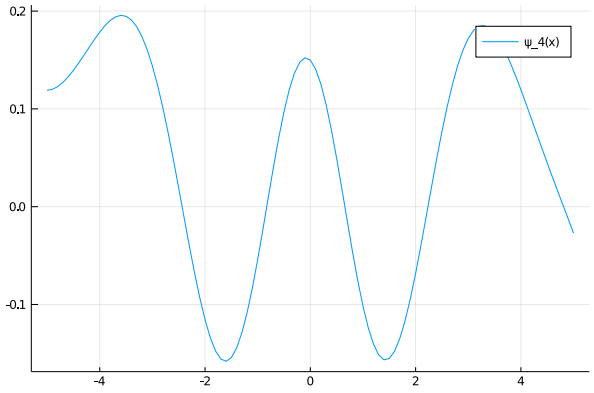

Figure 5: Graph of the wave function for the value of the quantum number $n = 4$

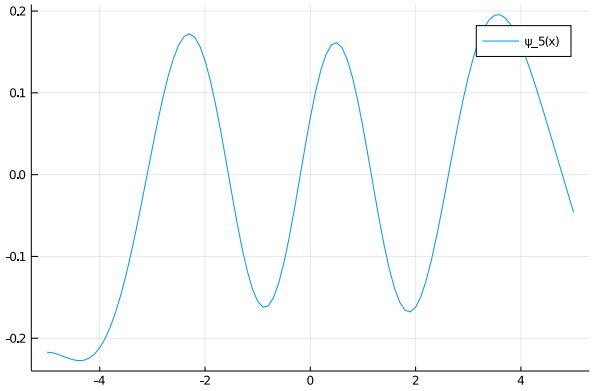

Figure 6: Graph of the wave function for the value of the quantum number $n = 5$

As a result of the scientific work the graphs and solutions were successfully obtained that correspond to the theoretical values of the behavior of the wave functions of a quantum harmonic oscillator. The developed program code turned out to be very compact, the development process was quick and convenient. The programming language Julia took upon itself libraries to solve the main problems considered during this research.

In summary, everything that has been said above, we have concluded that the language of Julia is fully applicable, and is very convenient in case of use as a tool for solving numerical modeling.

3. Conclusion

The research was executed to test the capabilities of the programming language Julia when it is used as a tool for experiments allowing to solve modern numerical problems in the field of numerical modeling.

As part of the task, the authors decided to consider the simplest problem of a quantum harmonic oscillator, using the most powerful library for solving ODEs of various kinds – `DifferentialEquations`. As a result of the scientific work, graphs and solutions corresponding to theoretical values were successfully obtained.

Acknowledgments

This work is supported by the Russian Science Foundation under grant 19-71-30008.

References

[1] L. D. Landau, E. M. Lifshitz, Mechanics, Course of Theoretical Physics. Vol. 1, 3rd ed., Butterworth-Heinemann, 1976.

[2] L. D. Landau, E. M. Lifshitz, Quantum Mechanics: Non-Relativistic Theory, volume 3, 3rd ed., Pergamon Press, 1977.

[3] J. Bezanson, S. Karpinski, V. B. Shah, A. Edelman, Julia: A Fast Dynamic Language for Technical Computing (2012) 1–27. arXiv:1209.5145.

[4] J. Bezanson, A. Edelman, S. Karpinski, V. B. Shah, Julia: A fresh approach to numerical computing, SIAM Review 59 (2017) 65–98. doi:10.1137/141000671. arXiv:1411.1607.

[5] J. Bezanson, J. Chen, B. Chung, S. Karpinski, V. B. Shah, J. Vitek, L. Zoubritzky, Julia: dynamism and performance reconciled by design, Proceedings of the ACM on Programming Languages 2 (2018) 1–23. doi:10.1145/3276490.

[6] M. N. Gevorkyan, A. V. Demidova, A. V. Korolkova, D. S. Kulyabov, Statistically significant performance testing of Julia scientific programming language, Journal of Physics: Conference Series 1205 (2019) 012017.1–7. doi:10.1088/1742-6596/1205/1/012017.

[7] C. Rackauckas, Q. Nie, DifferentialEquations.jl – A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia, Journal of Open Research Software 5 (2017). doi:10.5334/jors.151.

[8] The Julia Language, 2020. URL: https://julialang.org/.