Solutions of Schrödinger's Equation by Numerical Integration

Cameron Matchett

January 23, 2020

1 Abstract

In this project, numerical solutions for the energy eigenstates of the quantum harmonic oscillator are determined using the Numerov algorithm and are compared to the analytical solutions. Furthermore, a method is devised for determining the value of an energy eigenvalue given a trial value for $n = 0, 1, 2, 3$. The advantages and disadvantages of using the Numerov algorithm to find the numerical solutions are also discussed.

2 Introduction

The aim of this project was to obtain numerical solutions for the eigenfunctions of the quantum harmonic oscillator. For the potential associated with this problem it is straightforward to obtain the analytical solutions to the Schrödinger equation and so it is possible to compare these with the solutions obtained using the numerical method described below.

2.1 Basic Theory

The time-independent Schrödinger equation in one dimension is written as follows:

$$\frac{d^2\psi}{dx^2} + \frac{2m}{\hbar^2}[E - V(x)]\psi = 0 \tag{1}$$

We will examine numerical solutions for the case of the Quantum Harmonic Oscillator. In this case we have a potential of the form:

$$V(x) = \frac{1}{2}kx^2 \tag{2}$$

We can simplify the situation by introducing the dimensionless variables:

$$\hat{x} = \left(\frac{mk}{\hbar^2}\right)^{\frac{1}{4}} x \quad \text{and} \quad \hat{E} = \frac{2}{\hbar}\sqrt{\frac{m}{k}}E \tag{3}$$

Therefore we are left with the equation:

$$\frac{d^2\hat{x}}{d\hat{x}^2} = (\hat{V}(\hat{x}) - \hat{E})\psi \tag{4}$$

where $\hat{V}(\hat{x}) = \hat{x}^2$. We require that $|\psi|^2$ is integrable, and this only occurs when $\hat{E}$ is an energy eigenvalue of the Hamiltonian for the harmonic oscillator.

When this is the case, we have analytical solutions to equation (4) of the form:

$$\psi_n(\hat{x}) = H_n(\hat{x})e^{-\frac{\hat{x}^2}{2}} \tag{5}$$

where n is a non-negative integer and H_n are the Hermite polynomials. They are defined by $H_0 = 1$, $H_1 = 2\hat{x}$ and we have the following recurrence relation:

$$H_{n+1}(\hat{x}) = 2\hat{x}H_n(\hat{x}) - 2nH_{n-1}(\hat{x}) \tag{6}$$

The corresponding eigenvalues can be found using the relation:

$$\hat{E}_n = 2n + 1 \tag{7}$$

2.2 Numerical Method

We have a differential equation of the form:

$$\frac{d^2\psi}{d\hat{x}^2} = f(\hat{x})\psi \tag{8}$$

A suitable numerical method for solving such a differential equation is Numerov's Method. If we have a uniform grid with points separated by distance δ then the value of ψ at the grid point $j+1$ is related to the values at points $j-1$ and j by the following relation (see appendix section 7.1):

$$\left(1 - \frac{\delta^2}{12}f_{j+1}\right)\psi_{j+1} = \left(2 + \frac{5}{6}\delta^2 f_j\right)\psi_j - \left(1 - \frac{\delta^2}{12}f_{j-1}\right)\psi_{j-1} \tag{9}$$

where f_j and ψ_j are the values of f and ψ at $\hat{x} = j\delta$. The solutions for ψ fall into the two categories of even solutions and odd solutions. We can propose the following boundary conditions for each of the two cases:

$$\text{Even:} \qquad \psi(0) = 1 \qquad \frac{d\psi}{d\hat{x}}(0) = 0 \tag{10}$$

$$\text{Odd:} \qquad \psi(0) = 0 \qquad \frac{d\psi}{d\hat{x}}(0) = 1 \tag{11}$$

The above recurrence relation requires that we know the value of ψ at two different points before we can iterate to obtain our solution. The first value is given by one of the above boundary conditions, depending on whether n is even or odd. We can obtain the value of ψ at a second grid point by using a Taylor expansion. What we end up with is the following approximations to $\psi(\delta)$ for the even and odd cases:

$$\text{Even: } \psi(\delta) = \psi(0) + \frac{\delta^2}{2}f(0)\psi(0) + \frac{\delta^4}{24}[f''(0)\psi(0) + 2f'(0)\psi'(0) + f(0)^2\psi(0)] + ... \tag{12}$$

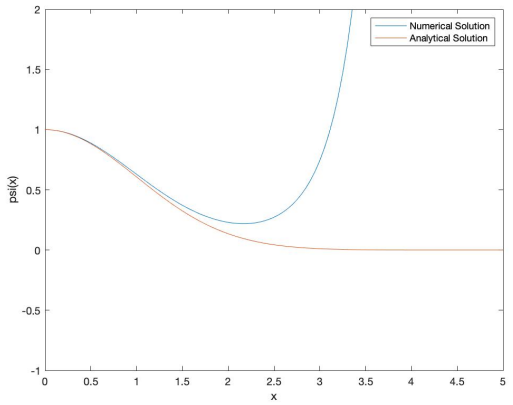
$$\text{Odd: } \psi(\delta) = \delta\psi'(0) + \frac{\delta^3}{6}[f(0)\psi'(0) + f'(0)\psi(0)] + ... \tag{13}$$

2.3 Iterative Method to Determine Energy Eigenvalues

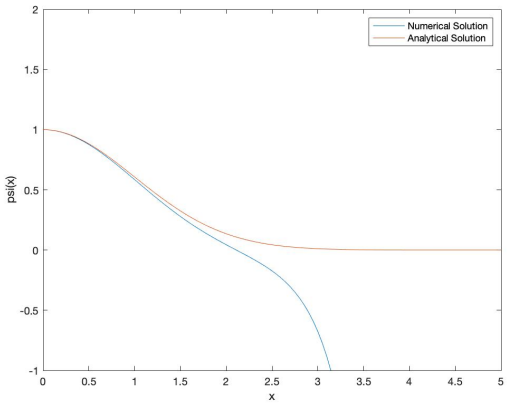
In the final part of the project, a method is used to determine the eigenvalues $\hat{E} = 1, 3, 5, 7$. A test value $\hat{E} = E_0$ is taken as an input into the function and the original function which provided the numerical solution to the Schrödinger equation is used to determine the solution for this particular energy value. Then, based on the value of n and whether the value of ψ at the largest $\hat{x}$ value is positive or negative, it is decided whether the value of $\hat{E}$ should be increased or decreased. The function then applies this difference to $\hat{E}$ until the value of $|\psi|$ at the largest $\hat{x}$ value is less than some threshold value. The value of $\hat{E}$ at this point gives the energy eigenvalue we are seeking. This is because we have (approximately) reached a point where $|\psi|^2$ is integrable.

3 Results

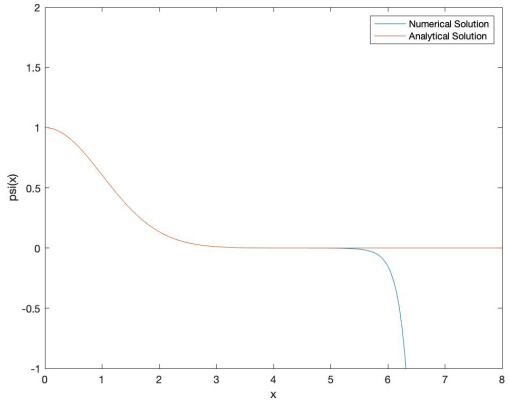
The following plots show the numerical and analytic solutions for the ground state wave function. For this analysis, we have $\delta = 0.05$ and $\hat{x}_1 = 5$. For the ground state, we have $n = 0$ and so the corresponding energy eigenvalue is $\hat{E} = 1$. The three plots show the difference in the numerical solution as $\hat{E}$ is varied in the vicinity of the known eigenvalue of 1. Plot (a) shows the result for $\hat{E} = 0.95$, plot (b) for $\hat{E} = 1.05$ and plot (c) for $\hat{E} = 1$.



(a) $\hat{E} = 0.95$



(b) $\hat{E} = 1.05$



(c) $\hat{E} = 1$

Figure 1: Numerical solutions for the different energy values.

For $\hat{E} = 1$ we have a very strong agreement between the numerical and analytical solutions for $\hat{x} = [0, 5]$. When we vary the value of $\hat{E}$ on either side of the known eigenvalue of 1, we see that the numerical solutions vary greatly from the analytical solution for large $\hat{x}$. The reason for this is that we are solving the time-independent Schrödinger equation and so we are seeking solutions for which the harmonic oscillator is in an energy eigenstate. This condition ensures that $|\psi|^2$ is integrable. When we vary the energy, it is no longer exactly equal to an eigenvalue of the Hamiltonian of the system and so we have a highly divergent solution whose modulus-square is not integrable. For $\hat{x} = [0, 5]$ the numerical and analytical solutions agree very well when $\hat{E}$ is exactly equal to the required energy eigenvalue of 1 for the stationary state. The reason for the disagreement at larger values of $\hat{x}$ is the fact that the local truncation errors (and hence global truncation errors) become very large at these $\hat{x}$ values. This is due to the fact that the function $f(\hat{x}) = \hat{x}^2 - \hat{E}$ increases greatly with $\hat{x}$. This increase in the global truncation error at large $\hat{x}$ results in a divergence of the numerical solution from the analytical solution.

4 Final Considerations

A brief outline of the derivation of the Numerov algorithm is given in appendix section 7.1. A major benefit of this algorithm is that it is more accurate than the widely-used fourth order Runge-Kutta method (RK4). It also requires less computation than RK4. However, the Numerov algorithm can only be applied to second order equations which either contain no first derivative term, or can be transformed in some way so as to remove their first derivative term. This means that RK4 is in general a more widely applicable method.

We could have used any of the Runge-Kutta methods to solve our original equation. These all vary in accuracy, but RK4 is the most accurate with the accumulated errors of $O(\delta^4)$ (which are still greater than the errors of $O(\delta^6)$ associated with the Numerov algorithm). We also could have used methods such as the Leap-Frog Method, Intrinsic Method or Predictor-Corrector Method. However, the success of the Numerov algorithm in providing a very accurate solution to a second order ODE with no first derivative term makes it a tough candidate to beat when it comes to solving the time-independent Schrödinger equation for the quantum harmonic oscillator.

5 Conclusion

In this investigation, we have used the Numerov method to determine numerical solutions to the time-independent Schrödinger equation for the quantum harmonic oscillator. It was discovered that the numerical solutions agreed very well with the analytical solutions when the energy $\hat{E}$ was set exactly equal to the known energy eigenvalue of the system for a given value of n. However these solutions diverged from the analytical solutions for large $\hat{x}$ due to the large local truncation errors associated with larger $\hat{x}$ values. The solutions diverged much sooner if the energy was varied on either side of the known energy eigenvalue for a given solution. A method was devised by which one could find the energy eigenvalues corresponding to $n = 0, 1, 2, 3$ given a trial value $\hat{E} = E_0$. The relative suitability of the Numerov method for solving the given ODE was evaluated and other options discussed, although the relatively high accuracy of the Numerov algorithm

and its applicability to second order ODEs with no first derivative term make it an ideal method.

6 References

1. EG, MK, CO22 'Solutions of Schrödinger's Equation by Numerical Integration', Oxford Physics, January 2018

2. AM, 'Computational Physics', http://www.cmth.ph.ic.ac.uk/people/a.mackinnon/Lectures/compphys/compphys.html, Imperial College London

3. PJS, 'The Numerov Algorithm', University of California, June 2001

7 Appendix

7.1 Outline of the Derivation of Numerov's Algorithm

We want to solve the following differential equation:

$$\frac{d^2\psi}{dx^2} - f(x)\psi = 0 \tag{14}$$

Consider the step size δ and act on the above differential equation with the operator: $1 + \frac{\delta^2}{12}\frac{d^2}{dx^2}$. This gives us the 'modified ODE':

$$\frac{\delta^2}{12}\frac{d^4\psi}{dx^4} - \frac{\delta^2}{12}\frac{d^2}{dx^2}[f(x)\psi] + \frac{d^2\psi}{dx^2} - f(x)\psi = 0 \tag{15}$$

We can express $\psi(x+\delta)$ and $\psi(x-\delta)$ in terms of their Taylor series about the point x. It is straightforward to show that some rearrangement of these Taylor series gives:

$$\psi''(x) = \frac{\psi(x+\delta) - 2\psi(x) + \psi(x-\delta)}{\delta^2} - \frac{\delta^2}{12}\psi''''(x) + O(\delta^6) \tag{16}$$

If we then plug in the expression for $\psi''(x)$ into the modified ODE, the fourth order derivatives sum to zero and we obtain:

$$-\frac{\delta^2}{12}\frac{d^2}{dx^2}[f(x)\psi] + \frac{\psi(x+\delta) - 2\psi(x) + \psi(x-\delta)}{\delta^2} - f(x)\psi = 0 \tag{17}$$

To handle the dependence of f on x in the above equation, we can use the following approximation which follows from (16):

$$\frac{d^2}{dx^2}[-f(x)\psi] \approx \frac{2f(x)\psi(x) - f(x+\delta)\psi(x+\delta) - f(x-\delta)\psi(x-\delta)}{\delta^2} \tag{18}$$

This leads to the Numerov algorithm as stated in the theory section at the beginning after some algebraic manipulation:

$$\psi(x+\delta) = \frac{2[1 + \frac{5}{12}\delta^2 f(x)] - [1 - \frac{\delta^2}{12}f(x-\delta)]\psi(x-\delta)}{1 - \frac{\delta^2}{12}f(x+\delta)} \tag{19}$$

7.2 Numerov Function

```
function result = numerov(x,psi,a,delta,f)
% Author: Cameron Matchett, Date: 22/09/2019
% Returns the Numerov iteration of the solution to d^2(psi)/dx^2 = f(x)*psi
% corresponding to index a.
% Input:
% * x: Array containing values of x in range x0 to x1 with separation
%      delta.
% * psi: Array containing the values of psi preceding the value being
%        calculated.
% * a: Index at which the iteraton is being calculated.
% * delta: Separatiopn of the x-values.
% * f: Lambda function which calculates the value of f(x) in the above
%      differential equation.
% Output:
% * result: Returns the next value of psi using the iteration.

% Using the formula for the Numerov iteration to obtain the next value of
% psi using the two previous values at inices (a-1) and (a-2).

result = (1/(1-((delta^2)/12)*f(x(a))))*((2+(5/6)*(delta^2)*f(x(a-1)))*
         psi(a-1)-(1-((delta^2)/12)*f(x(a-2)))*psi(a-2));

end
```

7.3 Function to solve Schrödinger Equation

```
function psi = solve_numerov(f,x,psi0,dpsi0,delta)
% Author: Cameron Matchett, Date: 22/09/2019
% This function solves d^2(psi)/dx^2 = f(x)*psi from x0 to x1 with boundary
% condition: psi(x0) = psi0 and (d(psi)/dx)(x0) = dpsi0.
% Input:
% * f: The function to be called on the right hand side of the equation.
%      This receives x and returns the value of f(x).
% * x: This is an array containing the values of x being integrated over.
%      First value in array is x0 and final value is x1.
% * psi0: The value of psi at x0.
% * dpsi0: The value of d(psi)/dx at x0.
%
% Output:
% * psi: An array containing the values of psi as calculated at each value
%        of x.
%
% Example use:
% >> f = @(x) x^2 ? 1;
% >> x = linspace(0, 1, 100);
% >> psi0=1;
% >> dpsi0 = 0;
% >> psi = solve_numerov(f, x, psi0, dpsi0);
% >> plot(x, psi);

% Creating an array containing the values of f(x):

f_array = [];
```

```
% Calculating the values of f for each value of x.

for a = 1:length(x)
    f_array = [f_array, f(x(a))];
end

% Using Taylor expansion to calculate a second value of psi to enable use
% of the Numerov iterative technique. If statement used to distinguish
% between the cases of odd and even solutions.

if psi0 == 0
    psi1 = delta*dpsi0 + ((delta^3)/6)*(f(0)*dpsi0);
else
    psi1 = psi0 + ((delta^2)/2)*f(0)*psi0 + ((delta^4)/24)*(2*psi0 + (f(0)^2)*psi0);
end

% Defining the array to hold the values of psi. Initially contains the
% value obtained from boundary condition and that from Taylor expansion.

psi = [psi0,psi1];

% For loop used to calculate the value of psi at index a using the Numerov
% iteration. The array psi is updated with each iteration to include the
% new value.

for a = 3:length(x)
    psi = [psi,numerov(x,psi,a,delta,f)];
end

end
```

7.4 Script for Solving Schrödinger Equation

```
% Author: Cameron Matchett, Date: 22/09/2019
% This script takes in start and end x-values, the step in x, the
% eigenvalue E and the principal quantum number n. It returns plots of the
% analytic solution to d^2(psi)/dx^2 = (x^2 - E)*psi as well as the
% numerical solution obtained using the Numerov method.

% Input variables:

delta = 0.05;
x0 = 0;
x1 = 5;
E = 1.05;
n = 0;

% Using if statement to decide boundary conditions for oscillator.

if mod(n,2) == 0
    psi0 = 1;
    dpsi0 = 0;
else
    psi0 = 0;
    dpsi0 = 1;
end
```

```
% Defining lambda function to calculate the value of the function
% multiplying psi in the differential equation above.

f = @(x) x^2 - E;

% Creating an array of the required x values.

x = x0:delta:x1;

% Using the solve_numerov function to obtain numerical solution of the
% differential equation stated above.

psi = solve_numerov(f,x,psi0,dpsi0,delta);

% Plotting the numerical and analytical solutions to the differential
% equation to enable comparison.

plot(x,psi);
xlim([0 5]);
ylim([-1 2]);
hold on
plot(x,exp(-0.5*x.^2));
legend('Numerical Solution','Analytical Solution');
xlabel('x')
ylabel('psi(x)');
```

7.5 Function to Find Energy Eigenvalues

```
function [E] = find_oscillator_eigenvalue(E0)
% Author: Cameron Matchett, Date: 22/09/2019
% Finding the eigenvalue for harmonic oscillator potential by iteration
% with starting value E0.
% Input:
% * E0: The initial guess of the eigenvalue.
% Output:
% * E: Value of the eigenvalue near the initial guess of the system with
%      harmonic oscillator potential.
% Example use:
% >> E = find_oscillator_eigenvalue(1.2);
% >> disp(E);

delta = 0.05;
x0 = 0;
x1 = 5;
n = 3;

% Defining initial conditions based on whether we have even or odd n.

if mod(n,2) == 0
    psi0 = 1;
    dpsi0 = 0;
else
    psi0 = 0;
    dpsi0 = 1;
end
```

```
E = E0;

f = @(x) x^2 - E;

x = x0:delta:x1;

psi1 = solve_numerov(f,x,psi0,dpsi0,delta);

% Determining the required direction of change in E by considering the sign
% of the wavefunction at the final x value.

if n < 2
    if psi1(101) > 0
        diff = 0.0001;
    else
        diff = -0.0001;
    end
else
    if psi1(101) > 0
        diff = -0.0001;
    else
        diff = 0.0001;
    end
end

% Using a while loop to carry out the iterative process of updating the
% value of E until the numerical solution closely matches the analytical
% solution.

while abs(psi1(101)) > 0.1

    E = E + diff;
    f = @(x) x^2 - E;
    psi1 = solve_numerov(f,x,psi0,dpsi0,delta);

end

E = round(E);

end
```