A Comparison of Popular Methods for the One-Dimensional Schrödinger Wave Equation

Luke Sianchuk[1], María Preciado[2], and Nick Dal Farra[3]

[1]20560770, ljsianch@uwaterloo.ca
[2]20914679, mrprecia@uwaterloo.ca
[3]20676841, ncdalfar@uwaterloo.ca

April 14, 2021

Abstract

The following report aims to replicate and extend the work presented by Reference [1]. The one-dimensional Schrödinger equation is simulated using the Crank-Nicolson, Leapfrog, and 4th order Runge-Kutta finite difference schemes which are analyzed across varying settings. Each numerical solver is quantitatively compared against known exact solutions of the harmonic oscillator and infinite square well problems in the investigation of L_1 errors, conservation of kinetic energy, conservation of probability, and computation time. The methods are also qualitatively compared for their ability to simulate the quantum tunneling effect.

Experiments revealed that all three methods were capable of exhibiting the quantum tunneling phenomenon, though each method differed in the other problem domains. The Crank-Nicolson scheme exhibited the best conservation of physical quantities and was always numerically stable. Next, the Leapfrog method proved to be the most efficient scheme in each problem setting and was usually stable. Finally, the RK4 solver provides the best accuracy for a particular choice of $\Delta x, \Delta t$, although it has the most restrictive stability region in terms of Δt.

Project code has been made available through the GitHub platform[1].

1 Introduction

The one-dimensional Schrödinger Equation (1) is a partial differential equation (PDE) that governs the time and space evolution of quantum systems in one dimension,

$$i\hbar\frac{\partial}{\partial t}\Psi(x,t) = \left[-\frac{\hbar}{2m}\frac{\partial^2}{\partial x^2} + V(x,t)\right]\Psi(x,t). \tag{1}$$

Ψ is a complex-valued function whose square modulus $|\Psi(x,t)|^2$ is the probability density of the quantum system's position. $\hbar$ is the angular Planck's constant, m is the system mass, and V describes the potential. V totally summarizes the environment's influence on the evolution of Ψ.

An extension of [1], the aim of this project is to answer the following question: when should a particular numerical method be chosen to simulate a quantum system? The answer to this question depends on factors including simulation context, computational resources, and the accuracy required of the results.

[1] GitHub repository: `https://github.com/nickdalfarra/schrodinger_simulation`

1.1 Simulations

In deriving analytical solutions to the upcoming problems, it is useful that $\Psi(x,t) = \psi(x)f(t)$ when the potential is independent of time [2, p. 20]. This allows exact solutions to be represented as a product of time dependent and independent functions. The MATLAB implementations of exact solutions make use of this representation to make code more modular, a good practice in programming.

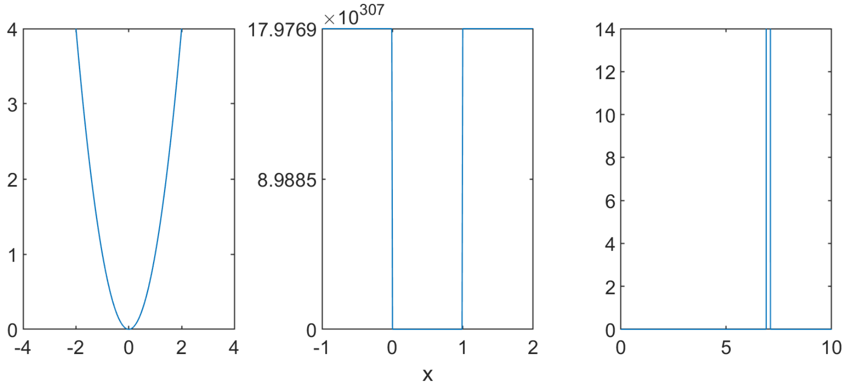


Figure 1: Potential functions $V(x)$ used in the simulations.

1.1.1 Harmonic Oscillator

The harmonic oscillator problem arises when potential is taken to be $V(x) \propto x^2$. The exact solution to this problem is defined in Equation 2.50 of [2, p. 35] by

$$\psi_n(x) = \left(\frac{m\omega}{\pi\hbar}\right)^{1/4} \frac{1}{\sqrt{2^n n!}} H_n(\xi) e^{-\xi^2/2}, \quad \xi = \sqrt{\frac{m\omega}{\hbar}}x$$

$$f(t) = e^{-iE_n t/\hbar}, \quad E_n = (n+1/2)\hbar\omega, \quad V(x) = \frac{1}{2}m\omega^2 x^2$$

where n is the discrete energy level, m is the mass, $\hbar$ is the angular Planck constant, ω is the angular frequency, and H_k is the k-th Hermite polynomial. Observe that the solution $\Psi = \psi f$ is periodic in time with period $\frac{2\pi\hbar}{E_n}$. ψ_n and f are implemented in `HO_ti.m` and `HO_td.m` respectively.

1.1.2 Infinite Square Well

The 1-D infinite square well problem simulates a particle confined to an interval of length l, trapped by infinite potential at each end. The environmental potential function $V(x)$ is given by

$$V(x) = \begin{cases} 0 & 0 \le x \le l \\ \infty & \text{otherwise} \end{cases}$$

This problem has stationary state solutions (Equation 2.31 in Griffiths) [2] given by

$$\Psi_n(x,t) = \sqrt{\frac{2}{l}} \sin\left(\frac{n\pi}{l}x\right) e^{-i\left(\frac{n^2\pi^2\hbar}{2ml^2}\right)t}$$

where l is the length of the well, n is the discrete energy level, $\hbar$ is the angular Planck constant, and m is the particle mass.

Notice that the above equation can be separated into its spatial and temporal components, given by

$$\Psi_n(x) = \sqrt{\frac{2}{l}} \sin\left(\frac{n\pi}{l}x\right)$$

and

$$\Psi_n(t) = e^{-i\left(\frac{n^2\pi^2\hbar}{2ml^2}\right)t}$$

respectively. These have been implemented in the MATLAB codes `SW_ti.m` (time independent) and `SW_td.m` (time dependent).

1.1.3 Quantum Tunneling: Narrow Gaussian

The following setup replicates an experiment performed by [1]. The final test is to shoot a free-moving particle at a high-potential barrier. A good choice of V is the probability density function (pdf) of a Gaussian distribution with mean $\mu = 7$ chosen ahead of the particle and a small variance of $\sigma^2 = 1 \times 10^{-4}$. This avoids discontinuity in the potential function (which can sometimes affect solver performance) and produces a sufficiently thin yet high-potential barrier. Taking $g(x \mid \mu, \sigma^2)$ to be the pdf of a Gaussian distribution with parameters μ, σ^2, we have

$$V(x) = g(x \mid 7, 10^{-4})$$

To construct a free particle that will move towards the potential barrier, we construct the initial condition as

$$\Psi(x, 0) = g(x \mid 6, 10^{-2}) \exp(-ik_0 x)$$

Justifying this construction is beyond the scope of the project; however it replicates the initial condition of an identical experiment in [1], and produces a wave function with the necessary property of moving the probability density rightward in a zero-potential environment.

There is no closed-form solution to this problem, precluding any error analysis on the quantum tunneling experiment. Also note that the initial condition is not normalized to have a total probability of one. This means that the results of any simulation would need to be normalized for the output to represent a true quantum system.

1.2 Methods

1.2.1 Crank-Nicolson

The Crank-Nicolson method is an implicit finite-difference method that is known for being numerically stable for any choice of $\Delta x, \Delta t$. For a one-dimensional PDE given by the formula

$$u_t = F(u, x, t, u_x, u_{xx}), \tag{2}$$

the Crank-Nicolson method is defined as being

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{1}{2}[F_j^{n+1} + F_j^n] \tag{3}$$

where $F_j^n = F(u^n, x^n, t_n, u_x^n, u_{xx}^n)$. This is effectively an arithmetic mean of the Forward and Backward Euler schemes. The Schrödinger equation can be written in the form of Equation (2) as

$$\frac{\partial \Psi}{\partial t} = i\frac{\hbar}{2m}\frac{\partial^2 \Psi}{\partial x^2} - i\frac{V}{\hbar}\Psi \tag{4}$$

thus making

$$F_j^n = i\left(\frac{\hbar}{2m}\frac{1}{\Delta x^2}\left(\Psi_{j-1}^n - 2\Psi_j^n + \Psi_{j+1}^n\right) - \frac{V_j^n}{\hbar}\Psi_j^n\right)$$

This leads to the Crank-Nicolson formulation of the 1D wave equation:

$$\begin{aligned}\frac{\Psi_j^{n+1} - \Psi_j^n}{\Delta t} &= i\frac{\hbar}{4m}\frac{1}{\Delta x^2}\left(\Psi_{j-1}^{n+1} - 2\Psi_j^{n+1} + \Psi_{j+1}^{n+1} + \Psi_{j-1}^n - 2\Psi_j^n + \Psi_{j+1}^n\right)\\ &- \frac{i}{2\hbar}\left(V_j^{n+1}\Psi_j^{n+1} + V_j^n\Psi_j^n\right)\end{aligned}$$

This implicit scheme requires solving a linear system on each time step of algorithm.

1.2.2 Leapfrog

The Leapfrog method is an explicit finite-difference scheme which uses central discretizations in both time and space. The first derivative in time of the Schrodinger equation is approximated as

$$\frac{\partial \Psi(x,t)}{\partial t} = \frac{\Psi(x, t+\Delta t) - \Psi(x, t-\Delta t)}{2\Delta t}$$

while the second derivative in space is estimated by

$$\frac{\partial^2 \Psi(x,t)}{\partial x^2} = \frac{\Psi(x+\Delta x, t) - 2\Psi(x,t) + \Psi(x-\Delta x, t)}{\Delta x^2}$$

Substituting these into the Schrodinger equation, we arrive at the Leapfrog approximation scheme given by

$$\frac{\Psi_j^{n+1} - \Psi_j^{n-1}}{2\Delta t} = i\frac{\hbar}{2m}\left(\frac{\Psi_{j+1}^n - 2\Psi_j^n + \Psi_{j-1}^n}{\Delta x^2}\right) - i\frac{V_j^n}{\hbar}\Psi_j^n$$

For implementation purposes this has been split into vectorized functions which return the real and imaginary components separately, as described by Berwick [3]. Rearranging the above equation to solve for the next time step and denoting the real and imaginary components by $R(\cdot)$ and $I(\cdot)$ respectively:

$$R(\Psi_j^{n+1}) = R(\Psi_j^{n-1}) - \frac{\Delta t}{\Delta x^2}(I(\Psi_{j+1}^n) - 2I(\Psi_j^n) + I(\Psi_{j-1}^n)) + 2\Delta t V_j^n I(\Psi_j^n)$$

$$I(\Psi_j^{n+1}) = I(\Psi_j^{n-1}) - \frac{\Delta t}{\Delta x^2}(R(\Psi_{j+1}^n) - 2R(\Psi_j^n) + R(\Psi_{j-1}^n)) + 2\Delta t V_j^n R(\Psi_j^n)$$

By definition, the Leapfrog method requires two previous time steps in order to operate. In the implementation here, this is achieved by approximating the first iteration using a forward in time and central in space finite difference method.

1.2.3 Classic Runge-Kutta

The classic or 4-th order Runge-Kutta method is also an explicit scheme and is used for the time derivative of the Schrödinger equation's solution. In this method, the slope of the solution at more than one point is used to calculate the next time step solution. We are interested in solving the initial value problem

$$\frac{\partial \Psi\left(x,t\right)}{\partial t} = -\frac{i}{\hbar}\left[-\frac{\hbar^2}{2m}\frac{\partial^2 \Psi\left(x,t\right)}{\partial x^2} + V(x,t)\Psi\left(x,t\right)\right]$$

with $\Psi(x,t_0) = \Psi_0(x)$. The second derivative with respect to space is calculated using a second-order central finite difference given by

$$\frac{\partial \Psi\left(x,t\right)}{\partial t} = -\frac{i}{\hbar}\left[-\frac{\hbar^2}{2m}\frac{\Psi\left(x+\Delta x,t\right) - 2\Psi\left(x,t\right) + \Psi\left(x-\Delta x,t\right)}{\Delta x^2} + V(x,t)\Psi\left(x,t\right)\right],$$

and this is the definition of the function f

$$\frac{\partial \Psi\left(x,t\right)}{\partial t} = f\left(\Psi\left(x,t\right),x,t\right).$$

The solution is thus calculated using RK4 as

$$\Psi(x,t+\Delta t) = \Psi(x,t) + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4),$$

where the four stages k are

$$\begin{aligned}
k_1 &= f(\Psi\left(x,t\right),x,t),\\
k_2 &= f\left(\Psi(x,t) + \frac{\Delta t}{2}k_1, x, t+\frac{\Delta t}{2}\right),\\
k_3 &= f\left(\Psi(x,t) + \frac{\Delta t}{2}k_2, x, t+\frac{\Delta t}{2}\right), \text{ and}\\
k_4 &= f\left(\Psi(x,t) + \Delta t k_3, x, t+\Delta t\right).
\end{aligned}$$

2 Test Framework

There are many reasons why a researcher may care to simulate the Schrödinger equation. For example, since Schrödinger's original modelling of the hydrogen atom in 1926 [4], no exact model of electrons for any other molecule have been discovered. No other molecules possess the same symmetry properties as single-proton hydrogen, making the general problem much more difficult. Furthermore, simulating the chemical bonds created by electrons is a central topic in the field of quantum chemistry.

This report analyzes numerical methods using quantitative and qualitative metrics. In addition to the regular numerical concerns of accuracy, stability, and efficiency, there are physical requirements in quantum physics that a good numerical method ought to meet.

A physical quantity of primary concern is total probability. Intuitively, the total probability of the spatial distribution for a quantum particle should be one at all times, i.e. $\forall t \int |\Psi(x,t)|^2 dx = 1$. This value is readily inferred during simulations by applying a functional grid norm:

$$\int |\Psi(x,t)|^2 dx \approx \sum_j |\Psi_j|^2 \Delta x = \Delta x \|\Psi\|^2$$

Similarly, the kinetic energy of a continuous quantum system can be inferred from a simulation.

Employing a central difference approach, we find that the kinetic energy $\langle T\rangle$ can be calculated as

$$\begin{aligned}\langle T\rangle &= -\frac{\hbar^2}{2m}\int \Psi^*\left(\frac{\partial^2}{\partial x^2}\right)\Psi dx \\ &\approx -\frac{\hbar^2}{2m}[\Psi^*]^T\cdot\frac{D}{\Delta x^2}\cdot[\Psi]\Delta x \\ \\ &= -\frac{\hbar^2}{2m\Delta x}[\Psi^*]^T\cdot D\cdot[\Psi] \qquad (5)\end{aligned}$$

where

$$D = \begin{pmatrix} -2 & 1 & & 1 \\ 1 & -2 & 1 & \\ & & \ddots & \\ 1 & & 1 & -2 \end{pmatrix}.$$

Here the notation $[\Psi]$ is used to indicate that we have changed from discussing the exact function Ψ to a vector containing approximations of Ψ along some finite and countable spatial support $\{x_j\}_{j=1}^{N}$.

Lastly, the numerical accuracy of each method is calculated using an L_1 grid-norm that has been modified slightly for compatibility with complex-valued functions:

$$e = \sum_j |\Psi_j - \Psi(x_j)|\, h$$

where $|\cdot|$ is the complex modulus, Ψ_j is the discretized solution and $\Psi(x_j)$ is the known exact solution evaluated at the appropriate spatial coordinate. These metrics are only useful when the exact Ψ function is known, which is true for the harmonic oscillator and square well problems. What about the qualitative tests mentioned earlier?

The idea of superimposed quantum states inspired Schrödinger to postulate “Schrödinger’s Cat”, a thought experiment in which a cat can only be accurately described as both alive and dead until its survival has been measured. While he proposed this idea to demonstrate the absurdity of his fundamentally probabilistic quantum mechanics, good numerical solvers must replicate this “absurd” phenomenon to accurately represent quantum behaviour. In the quantum tunneling problem, the group hopes to observe leaking of the system’s positional distribution through the potential barrier.

3 Results and Discussion

3.1 Parameters for Simulations

In Table 1, we show the parameters used in the simulations we present herein. We specify the spatial domain, the number of grid points N, the spatial step size Δx, the simulation final time t_{end} and the number of iterations that were used consistently for all three methods. N was selected to ensure smoothness of the approximated wave-function. For example, in the free particle problem $N = 1400$ to resolve the oscillations in the solution. On the other hand, the time step size chose Δt depended on Δx and t_{end} to ensure convergence of the solution produced by the RK4 method.

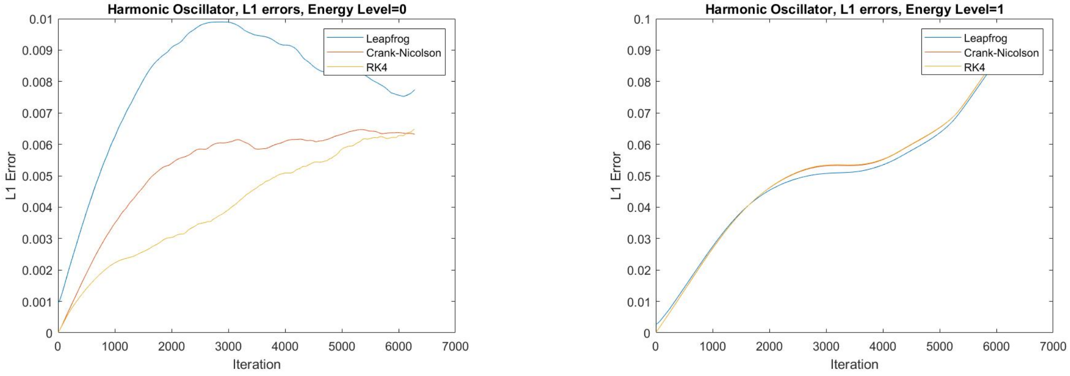
Table 1: Parameters used for all the methods specified by problem or potential. N is the number of grid points, Δx is the spatial step size, t_{end} is the final time of the simulation, Δt is the time step size, and the last column shows the number of iterations.

Problem	Domain	N	Δx	t_{end}	Δt	Iterations
Harmonic oscillator	[-4,4]	50	0.16	3.1416	5×10^{-4}	6284
Square Well	[0,1]	50	0.02	3.1416	7×10^{-7}	4487989
Free Particle Tunneling	[0,14]	1400	0.01	0.1	1×10^{-6}	1×10^{5}

3.2 L_1 Simulation Error

The following plots illustrate the L_1 errors of each method as a function of the iteration number.

Figure 2: L_1 Errors in the Quantum Harmonic Oscillator Problem

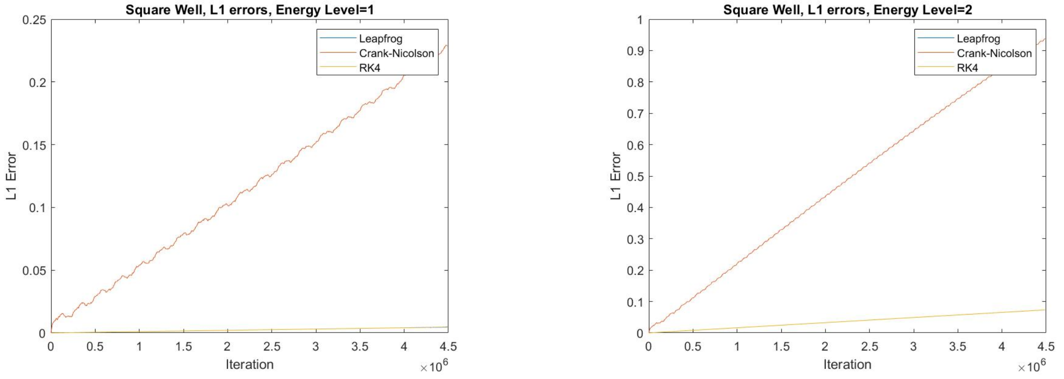


(a) Energy Level n=0

(b) Energy Level n=1

Looking at the harmonic oscillator problem, there is marginal variation in the L_1 errors between each of the three methods throughout the iterations. At the first energy level, $n = 0$, it is evident that the Leapfrog scheme performs slightly worse than the Crank-Nicolson and RK4 methods, though there remains comparable errors relative to one another. These relative differences are nearly indiscernible at the next energy level, $n = 1$.

Figure 3: L_1 Errors in the Infinte Square Well Problem



(a) Energy Level n=1

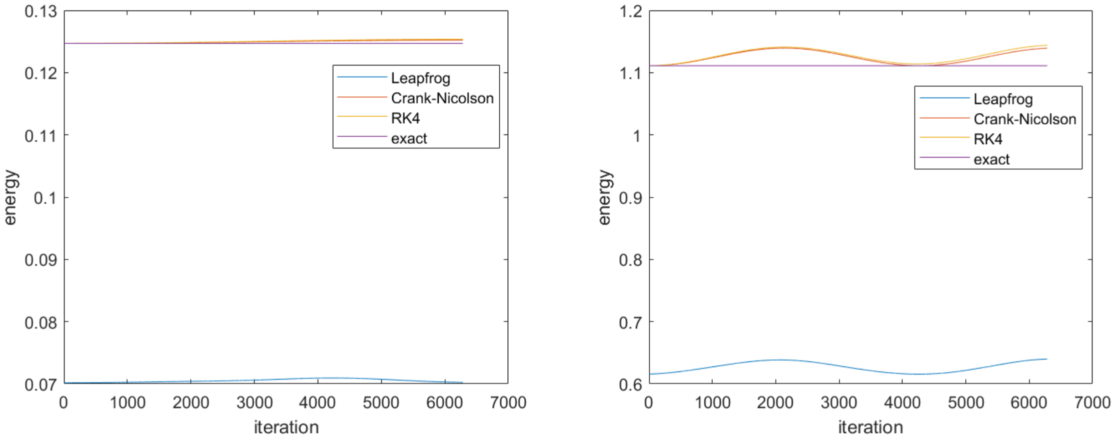
(b) Energy Level n=2

In contrast, the Leapfrog and RK4 errors plot with near identical values, both greatly outperforming the Crank-Nicolson scheme in the infinite square well problem. The approximation of infinity defined for this simulation causes the matrix used in the Crank-Nicolson method to have a condition number of approximately 10^{33}. This extreme value creates instability when solving the linear system in MATLAB, causing much greater error than what is seen in the other two methods.

3.3 Kinetic Energy

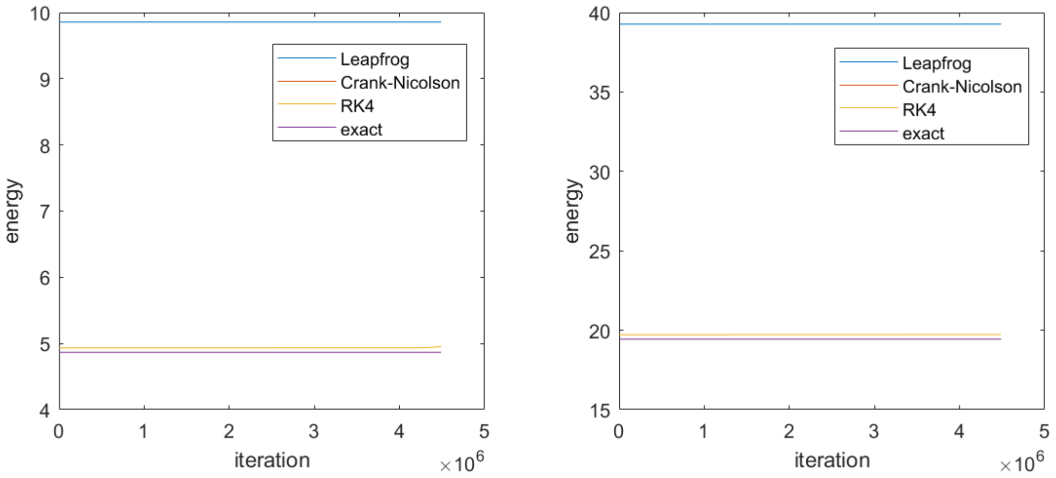
The following plots demonstrate the inferred energy values calculated using Equation 5.

Figure 4: Inferred energy values for the Harmonic Oscillator problem for energy levels $n = 0, 1$.



For the harmonic oscillator, a sinusoidal behaviour can be observed in the energy. This phenomenon occurs for all methods. Clearly the energy of the exact solution is constant. Through additional experimentation, it appeared that Δx played a role in the amplitude of these oscillations waves. Amplitude was independent of the simulation radius (maximum $|x|$ value simulated) beyond a sufficiently large radius, indicating that the phenomenon was not due to the periodic boundary conditions. The team was ultimately unable to explain this effect.

Figure 5: Inferred energy values for the Square Well problem for energy levels $n = 1, 2$. Note that in both plots, the Crank-Nicolson line is covered over by the exact solution.

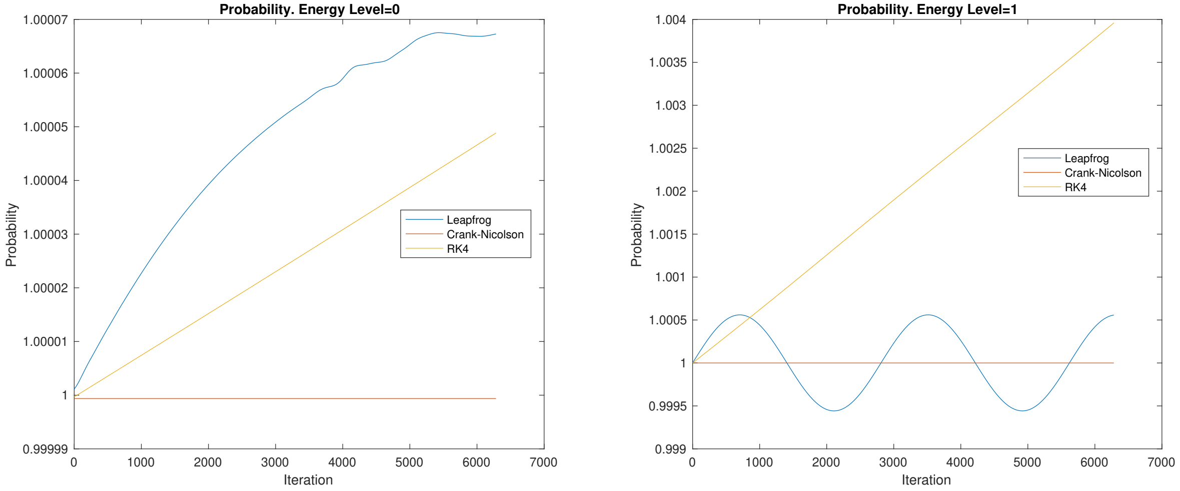


Across all experiments, kinetic energy inferred from the Leapfrog method is very different than the exact kinetic energy (relative error ranges from approximately $50 - 100\%$). This difference exists on the very first iteration of the algorithm. This suggests that the large discrepancy is due to the initial Euler step used in the algorithm, which is generally known to have low-order accuracy. This difference could be altogether eliminated by using a more accurate numerical solver to perform the first time step in Leapfrog, although further testing would be required to confirm this hypothesis.

3.4 Probability Conservation

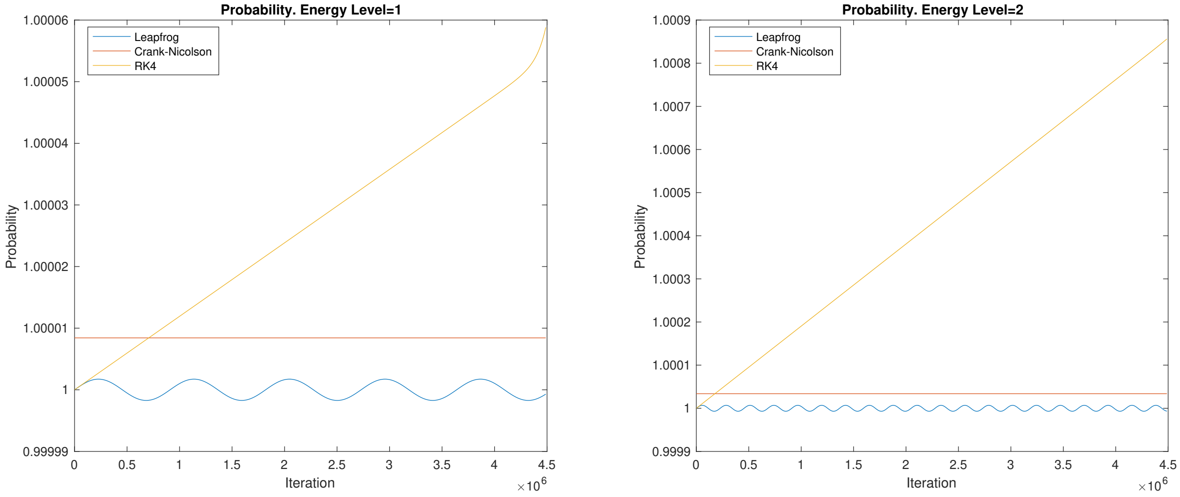
Another way to assess the methods is to check if the probability is conserved. The integral over the whole domain of the absolute value squared of the wavefunction is expected to be one in order to guarantee the physical meaning of the wavefunction. In Figure 6, we plot such probability as a function of time, or the number of iterations, for the Harmonic Oscillator potential. In both the $n = 0$ and $n = 1$ energy levels, the Crank-Nicolson method yielded a constant value of the probability at one. The probability from the RK4 method slowly increases with time, reaching increments of 5×10^{-5} and 4×10^{-3} for $n = 0$ and $n = 1$, respectively. The Leapfrog scheme produced values of the probability that oscillate with time but that are bounded by $1 \pm 7 \times 10^{-5}$ and $1 \pm 6 \times 10^{-4}$ for $n = 0$ and $n = 2$, approximately. Crank-Nicolson excels the other methods on conserving the probability. Nevertheless, the other methods presented shifts that were very small compared to the probability's expected value.

Figure 6: Probability density integrated over the whole domain as a function of time for the Harmonic Oscillator for energy levels $n = 0, 1$.



In Figure 7, we show the probability as a function of time for the Infinite Square Well for the $n = 1$ and $n = 2$ energy levels. As observed for the Harmonic Oscillator, the Crank-Nicolson method produces a probability value that is constant in time for both $n = 1$ and $n = 2$ energy levels. The RK4 method also yields a slowly increasing probability, and the Leapfrog scheme's probability oscillates with time for both energy levels. The shifts for RK4 and Leapfrog are again minimal: $1 + 6 \times 10^{-5}$ and $1 \pm 2 \times 10^{-6}$ for $n = 1$, and $1 + 9 \times 10^{-4}$ and $1 \pm 1 \times 10^{-5}$ for $n = 2$. As observed, the oscillations in the probability by the Leapfrog have an increased frequency when increasing the energy level. For the Infinite Square Well, the Crank-Nicolson method also conserves the probability better when compared to the other methods.

Figure 7: Probability density integrated over the whole domain as a function of time for the Infinite Square Well for energy levels $n = 1, 2$.



Perhaps it is not surprising that Crank-Nicolson preserves total probability. In deriving the numerical method, we find that Crank-Nicolson defines an expression of the form

$$A\Psi^{t+1} = B\Psi^t$$

where A and B are unitary matrices [5, p. 65]. If U is unitary then by definition $U^{-1} = U^*$. If U is unitary then U^{-1} is also unitary; this is easy to show, since $(U^{-1})^* = (U^*)^* = U = (U^{-1})^{-1}$. Also note that a product of unitary matrices U and V is also unitary: we have $(UV)^*UV = I = UV(UV)^*$. The last useful property of unitary matrices is that they are norm-preserving; that is, $\|Uv\| = \|v\|$ for all $v \in \mathbb{C}^n$. This is true because $\|Uv\|^2 = v^*U^*Uv = v^*v = \|v\|^2$.

Armed with these facts, we can see that $\Psi^{t+1} = U\Psi^t$ for the unitary matrix $U = A^{-1}B$. Since the total system probability is calculated using the expression $\Delta x\|\Psi\|^2$, we have

$$\|\Psi^{t+1}\|^2 = \|U\Psi^t\|^2 = \|\Psi^t\|^2 \tag{6}$$

Therefore we have proven that the numerical Crank-Nicolson scheme preserves total system probability on every iteration of the algorithm, providing a theoretical grounding for the notably excellent conservation properties of the Crank-Nicolson scheme.

3.5 Work-Δt Relationship

For this experiment, $N = \text{len}(x) = 50$, `t_end` $= 2\pi$ are chosen for the harmonic oscillator problem. Execution time is recorded for $\Delta t = 0.1$, reduced by a constant factor, and so on until $\Delta t \approx 10^{-4}$. The experiment was performed on a Dell Latitude E5470 laptop with an Intel i5-6200U CPU @ 2.30GHz (2 cores) with 8 GB of RAM. All applications other than MATLAB version R2019b were closed, although Windows 10 background processes and other daemons were not controlled for.

Figure 8 demonstrates the relationship between computational work and choice of Δt.

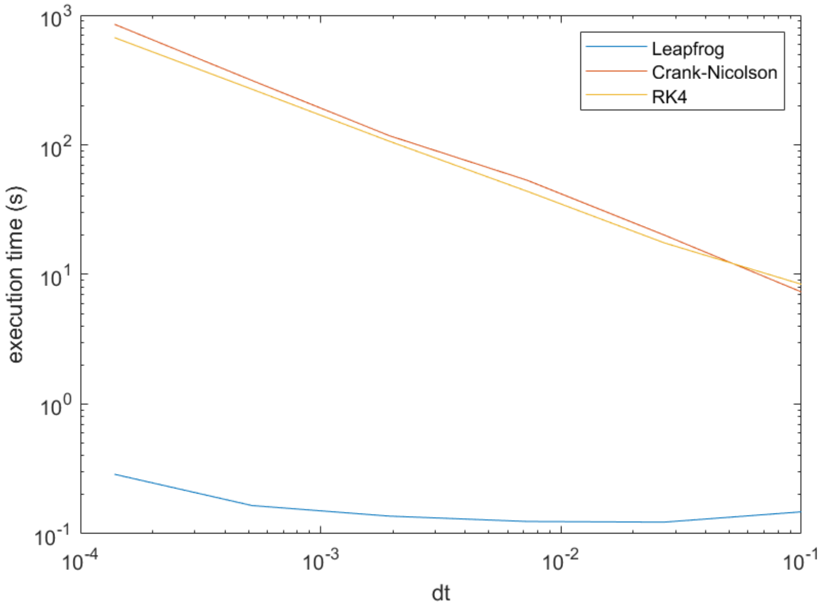


Figure 8: Log-log plot of computational work versus choice of Δt. A linear relationship on a log-log plot indicates an exponential relationship.

A straight line on logarithmic axes indicates an exponential relationship, and we observe this connection between Δt and work for both the RK4 and Crank-Nicolson methods. An exponential relationship is not surprising; if N_{iter} is the number of time steps, T is the time length of the simulation and $T \gg \Delta t$, then $N_{\text{iter}} \approx \frac{T}{\Delta t}$. Assuming work ($W$, measured in seconds) is proportional to N_{iter}, then $\exists c \in \mathbb{R} : W = cN_{\text{iter}}$. Then

$$\log W = \log \frac{cT}{\Delta t} = -\log \Delta t + \log cT$$

This is not quite the relationship observed for RK4 and Crank-Nicolson; by fitting a straight line the the endpoints of the RK4 curve, the work-Δt relationship for both RK4 and CN was found to be approximately

$$W = 1.5\,(\Delta t)^{-0.7}$$

The fact that Δt's exponent is not -1 indicates that optimization is taking place as the number of iterations increases. It is speculated that this is caused by MATLAB resource management protocols.

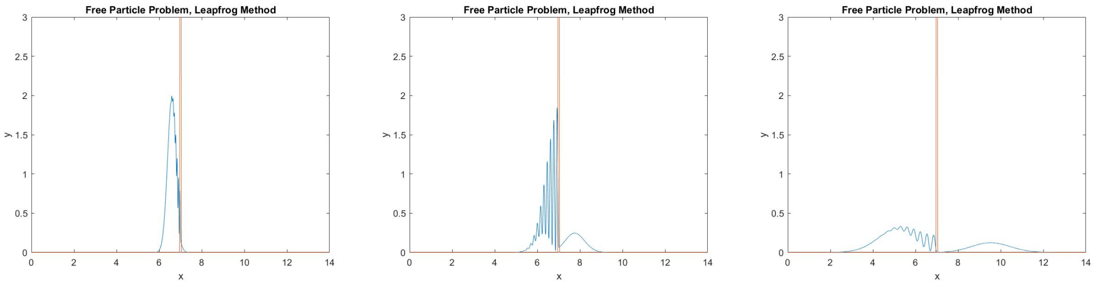
A surprising result is the radical difference in execution time for Leapfrog. For $\Delta t \approx 10^{-4}$, Leapfrog is over three orders of magnitude faster than the other two methods. The group speculates that Leapfrog's speed is due to its highly vectorized implementation; all the calculations used in updates are expressed as vector operations for which MATLAB is highly optimized.

The strange behaviour where Leapfrog execution time *decreases* going from $\Delta t = 10^{-1}$ to 10^{-2} is speculated to be due to the same reason that the work relationship for RK4 has an exponent of -0.7 instead of -1; allocation of resources are dynamic during a script's execution, leading to speedup that wouldn't be expected if resources were optimally allocated at the beginning of the script. This discrepancy may also be due to background process noise on the laptop, which would have a greater relative influence on Leapfrog's execution time than the other methods.

3.6 Quantum Tunneling Behaviour

The following plots demonstrate the quantum tunneling effect for each numerical method at selected time steps.

Figure 9: Leapfrog Simulation of the Free Particle Tunneling Problem

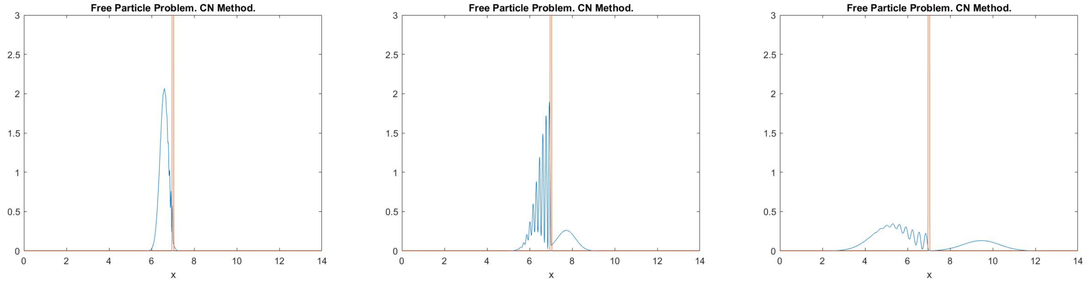


(a) time = 0.02

(b) time = 0.05

(c) time = 0.1

Figure 10: Crank-Nicolson Simulation of the Free Particle Tunneling Problem

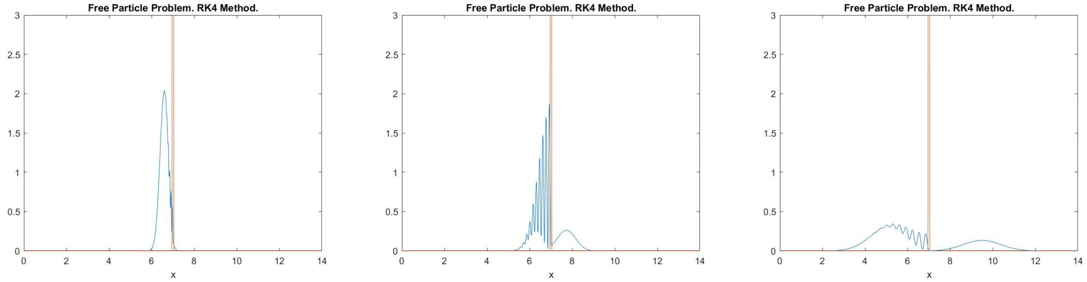


(a) time = 0.02

(b) time = 0.05

(c) time = 0.1

Figure 11: RK4 Simulation of the Free Particle Tunneling Problem



(a) time = 0.02

(b) time = 0.05

(c) time = 0.1

Without a known exact solution, the quantum tunneling problem is investigated by qualitative means only. Using each of the three numerical schemes, a snapshot is taken at specific time intervals for comparison. In each case we see the probability distribution, drawn by the blue wave, begin by travelling to the right. As the wave reaches the orange potential barrier, oscillations arise. Then with the progression of time, some of the probability distribution continues to the right of the barrier, while some is reflected back to the left. Looking vertically through the combination of the above figures, it is evident that each scheme produces very similar results. Thus, the quantum tunneling effect is reasonably simulated by each of the three methods.

4 Conclusion

4.1 Findings

Each of the three numerical schemes exhibited unique performances characteristics. Error analysis revealed poor results from the Crank-Nicolson scheme in the infinite square well problem; the extremely high potential creates a matrix with a condition number on order 10^{33}, resulting in severe numerical instability when solving for Ψ^{t+1} using Ψ^{t}.

Kinetic energy conservation showed similar results for the Crank-Nicolson and RK4 schemes, while the Leapfrog method exhibited relative errors of 50%-100%. It is speculated that this error is caused by the initial Euler step in this implementation. Probability conservation experiments revealed an excellent performance by the Crank-Nicolson scheme which was theoretically proven to conserve total probability in Equation 6. The Leapfrog method displays some oscillation about total probability 1, while the RK4 scheme shows increasingly inaccurate results as time progresses.

An examination of the work-Δt connection showed an unsurprising exponential relationship for the RK4 and Crank Nicolson methods. The Leapfrog method exhibited excellent computational efficiency which can be attributed to its highly vectorized implementation. Finally, each of the three numerical schemes showed qualitatively comparable results in the potential barrier simulations, each exhibiting the quantum tunneling effect.

Now we can finally answer the question posed in the introduction: when should a particular numerical method be chosen to simulate a quantum system? Firstly, the Crank-Nicolson scheme proves to be a versatile method with excellent performance in the conservation of physical quantities. Next, the Leapfrog method should be used for efficient computation in a variety of problem settings. Lastly, the RK4 method should be used when numerical accuracy for a particular choice of $\Delta x, \Delta t$ is the most important outcome.

4.2 Future Direction

This project leaves some promising avenues of research to be explored. Firstly, having theoretical stability conditions from Von Neumann analysis would be allow researchers to predict a priori whether or not a numerical solutions will converge. Without these conditions, experimentation on Δt (which depends on the choice of Δx) consumes a larger portion of the research process. Note that stability conditions for RK solvers tend to be defined by quartic curves, whereas Leapfrog stability tends to be a linear inequality and Crank-Nicolson stable for all choices of $\Delta t, \Delta x$.

Another direction for future research is exploring the conservation of other physical quantities, such as momentum and quantum potential energy. Finally, it would be interesting to experiment on numerical properties for problems with time-dependent potential; would this influence the accuracy of each method? One example of a time-varying potential is electromagnetic interaction with a laser pulse: $V(t) = A\cos(\omega_L t)$ where A is the amplitude of the laser wave and ω_L its frequency. This would require changes to the definition of the potential and the k stages of the RK4 method, but would be relatively straightforward to implement.

References

[1] Loren Jørgensen, David C. Lopes, and Etienne Thibierge. Numerical resolution of the schrödinger equation. Numerical analysis project by Masters students of École Normale Supérieure de Lyon in France.

[2] David J. Griffiths. *Introduction to Quantum Mechanics.* Prentice Hall, Inc., New Jersey, 1995.

[3] Kevin Berwick. *Computational Physics using MATLAB.* Purdue, 2012.

[4] Erwin Rudolf Josef Alexander Schrödinger. An undulatory theory of the mechanics of atoms and molecules. *The Physical Review*, 28, December 1926.

[5] Marvin Q. Jones. Numerical methods in quantum mechanics: Analysis of numerical schemes on one-dimensional schrödinger wave problems. Master's thesis, North Carolina AT State University, 2013.