Crank-Nicolson Method for Partial Differential Equations

Elijan Jakob Mastnak
Student ID: 28181157

December 2020

Contents

Assignment

1. Analyze the time evolution of the initial state

$$\psi(x,0) = \frac{\alpha^{1/2}}{\pi^{1/4}} e^{-\frac{\alpha^2}{2}(x-a)^2}$$

in the harmonic potential $V(x) = \frac{kx^2}{2}$, where $k = \omega^2$ and $\alpha = k^{1/4}$ and .

Use the parameters $\omega = 0.2$ and $a = 10$. Use a position grid $x \in [a, b] = [-40, 40]$ with $N = 300$ points. Choose a time step Δt suited to the oscillatory period $T = \frac{2\pi}{\omega}$. Observe the time evolution over ten oscillatory periods.

2. Analyze the time evolution of the initial Gaussian wave packet

$$\psi(x,0) = (2\pi\sigma_0^2)^{-1/4} e^{ik_0(x-a)} e^{-(x-a)^2/(2\sigma_0)^2}$$

in free space with $V = 0$. Use the parameters $\sigma_0 = \frac{1}{20}$, $k_0 = 50\pi$, $a = 0.25$.

Use the position values $x \in [-0.5, 1.5]$ and a time step $\Delta t = 2\Delta x^2$. Observe the time evolution until the wave packet's center reaches $x \approx 0.75$.

3. *Optional*: Solve the same problems with a higher-order approximation of the position and time derivatives.

Before flooding you with 9 pages of theory, here is a friendly, colorful picture:

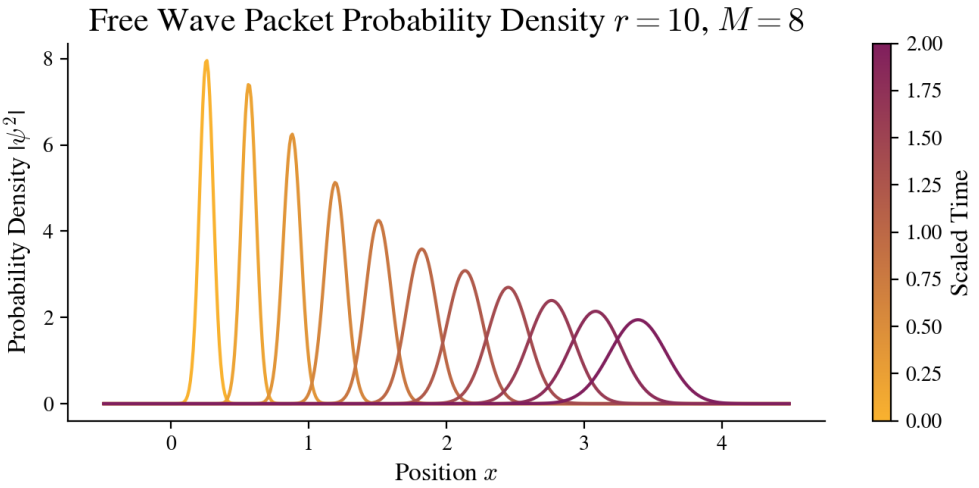

Figure 1: A free wave packet's probability density traveling along the x axis.

1 Theory

To skip the introductory theory, see Section 2. To skip all theory completely and jump right to the results, see Section 4.

This report involves solving the time-dependent Schrödinger equation

$$\left(i\hbar\frac{\partial}{\partial t} - H\right)\psi(x,t) = 0$$

with a time-independent Hamiltonian operator of the form

$$H = -\frac{\hbar}{2m}\frac{\partial^2}{\partial x^2} + V(x)$$

For numerical computation, it is more convenient to work in natural units. To do so, we introduce the change of variables

$$\frac{H}{\hbar} \to H, \qquad x\sqrt{\frac{m}{\hbar}} \to x, \qquad \frac{1}{h}V\left(x\sqrt{\frac{m}{\hbar}}\right) \to V(x)$$

This change of variables effectively sets $\hbar = m = 1$ and produces the Hamiltonian

$$H = -\frac{1}{2}\frac{\partial^2}{\partial x^2} + V(x) \tag{1}$$

1.1 Basic Crank-Nicolson Solution

We approximate the time evolution of the state $\psi(x,t)$ to $\psi(x,t+\Delta t)$ with a Taylor approximation of the time evolution operator $e^{-H\Delta t}$

$$\psi(x,t+\Delta t) = e^{-iH\Delta t}\psi(x,t) \approx \frac{1-\frac{1}{2}iH\Delta t}{1+\frac{1}{2}iH\Delta t}\psi(x,t) \tag{2}$$

This approximation is unitary, with an error of order $\mathcal{O}(\Delta t^3)$.
First, we partition the x interval $[x_0, x_J]$ into a grid of $J+1$ points $\{x_j\}_0^J$ separated by the uniform step size

$$\Delta x = \frac{x_j - x_0}{J-1} \implies x_j = x_0 + j\Delta x, \quad j = 0,1,\ldots,J-1$$

We solve the Schrödinger equation for $t \in [t_0, t_N]$, and find the discrete time points with $t_n = t_0 + n\Delta t$ for $n = 0,1,\ldots,N$. We approximate the second position derivative $\frac{\partial^2}{\partial x^2}$ with a finite difference approximation

$$\frac{\partial^2\psi}{\partial x^2} \approx \frac{\psi(x+\Delta x,t) - 2\psi(x,t) + \psi(x-\Delta x,t)}{\Delta x^2} \equiv \frac{\psi_{j+1}^n - 2\psi_j^n + \psi_{j-1}^n}{\Delta x^2}$$

where $V(x_j) = V_j$ and $\psi(x_j,t_n) \equiv \psi_j^n$. Substituting this expression into the approximate Schrödinger equation (Eq. 2) and writing the Hamiltonian in the dimensionless form of Equation 1 produces the system of equations

$$\begin{aligned}\psi_j^{n+1} - i\frac{\Delta t}{4\Delta x^2}\left[\psi_{j+1}^{n+1} - 2\psi_j^{n+1} + \psi_{j-1}^{n+1}\right] + i\frac{\Delta t}{2}V_j\psi_j^{n+1} \\ = \psi_j^n + i\frac{\Delta t}{4\Delta x^2}\left[\psi_{j+1}^n - 2\psi_j^n + \psi_{j-1}^n\right] - i\frac{\Delta}{2}V_j\psi_j^n\end{aligned}$$

We assume homogeneous boundary conditions and set $\psi_j^n = 0$ for $j < 0$ and $j > N$. We then introduce the vector

$$\boldsymbol{\psi}^n = \left[\psi_0^n, \psi_1^n, \ldots, \psi_J^n\right]^T \in \mathbb{C}^{J+1},$$

and write the system of equations as a matrix:

$$\mathbf{A}\boldsymbol{\psi}^{n+1} = \mathbf{A}^*\boldsymbol{\psi}^n, \quad \text{where} \quad \mathbf{A} = \begin{pmatrix} d_0 & a & & & & \\ a & d_1 & a & & & \\ & a & d_2 & a & & \\ & & \ddots & \ddots & \ddots & \\ & & & a & d_{J-1} & a \\ & & & & a & d_J \end{pmatrix} \in \mathbb{C}^{J+1\times J+1}$$

where

$$b = i\frac{\Delta t}{2\Delta x^2}, \qquad a = -\frac{b}{2}, \qquad d_j = 1 + b + i\frac{\Delta t}{2}V_j$$

Finding $\psi(x,t)$ thus reduces to repeatedly solving the matrix equation

$$\mathbf{A}\boldsymbol{\psi}^{n+1} = \mathbf{A}^*\boldsymbol{\psi}^n$$

1.2 Higher-Order Position Approximation

Note that the following comes directly from [1]. I have retyped it only to solidify my own understanding, but the work is wholly unoriginal. For a higher-order position approximation, we discretize the second spatial derivative according to

$$y''(x) = \frac{1}{h^2}\sum_{k=-r}^{r} c_k^{(r)} y(x+kh) + \mathcal{O}(h^{2r})$$

The $c_k^{(r)}$ are real constants coming from the Taylor expansions of $y(x+kh)$ and $y(x-kh)$ and satisfy $c_{-k}^{(r)} = c_k^{(r)}$ for $k = 1, 2, \ldots, r$. See e.g. Equations 2.6 to 2.8 of [1] for a thorough discussion. Table 1 shows the constants $c_k^{(r)}$ for $r = 1, \ldots, 7$; see [2] for an algorithm to calculate the coefficients for arbitrarily large n.

r	$k=0$	1	2	3	4	5	6	7
1	-2	1						
2	$-\frac{5}{2}$	$\frac{4}{3}$	$-\frac{1}{12}$					
3	$-\frac{49}{18}$	$\frac{3}{2}$	$-\frac{3}{20}$	$\frac{1}{90}$				
4	$-\frac{205}{72}$	$\frac{8}{5}$	$-\frac{1}{5}$	$\frac{8}{315}$	$-\frac{1}{560}$			
5	$-\frac{5269}{1800}$	$\frac{5}{3}$	$-\frac{5}{21}$	$\frac{5}{126}$	$-\frac{5}{1008}$	$\frac{1}{3150}$		
6	$-\frac{5369}{1800}$	$\frac{12}{7}$	$-\frac{15}{56}$	$\frac{10}{189}$	$-\frac{1}{112}$	$\frac{2}{1925}$	$-\frac{1}{16632}$	
7	$-\frac{266681}{88200}$	$\frac{7}{4}$	$-\frac{7}{24}$	$\frac{7}{108}$	$-\frac{7}{528}$	$\frac{7}{3300}$	$-\frac{7}{30888}$	$\frac{1}{84084}$

Table 1: Coefficients $c_k^{(r)}$ for $r = 1, \ldots, 7$.

As in the basic CN scheme, partition the time and position values according to

$$x_j = x_0 + j\Delta x, \qquad j = 0, 1, \ldots, J$$
$$t_n = t_0 + n\Delta t, \qquad n = 0, 1, \ldots, N$$

Next, we insert the generalized finite difference approximation for $y''(x)$ into approximate unitary Schrödinger equation to get the system of equations

$$\psi_{j,n+1} - \frac{i\hbar\Delta t}{4m(\Delta x)^2}\left[\sum_{k=-r}^{r} c_k^{(r)}\psi(j+k,n+1)\right] + \frac{i\Delta t}{2\hbar}V_j\psi_{j,n+1}$$
$$= \psi_{j,n} + \frac{i\hbar\Delta t}{4m(\Delta x)^2}\left[\sum_{k=-r}^{r} c_k^{(r)}\psi(j+k,n)\right] - \frac{i\Delta t}{2\hbar}V_j\psi_{j,n}$$

for $j = 0, 1, \ldots J$. For homogeneous boundaries, we set $\psi_{j,n} = 0$ for $j < 0$ and $j > J$. To simplify the system of equations, we define the auxiliary quantities

$$b \equiv \frac{i\hbar\Delta t}{2m(\Delta x)^2}, \quad z_1^{(1)} \equiv -2, \quad a_k^{(r)} \equiv \frac{b}{z_1^{(1)}}c_k^{(r)} \tag{3}$$

where the $c_k(r)$ are defined in Table 1. In terms of $a_k^{(r)}$, b and $z_1^{(1)}$, we define d_j as

$$d_j = 1 + a_0^{(r)} - \frac{i\Delta t}{\hbar}\frac{V_j}{z_1^{(1)}}, \quad j = 0, 1, \ldots, J \tag{4}$$

The solution for the wavefunction $\psi_{j,n+1}$ comes from solving

$$\mathbf{A}\boldsymbol{\psi}_{n+1} = \mathbf{A}^*\boldsymbol{\psi}_n, \quad n = 0, 1, \ldots N-1 \tag{5}$$

where $\boldsymbol{\psi}_n = \left[\psi_{0,n}, \psi_{1,n}, \ldots, \psi_{J,n}\right]^T \in \mathbb{C}^{J+1}$ is the system's wavefunction at time t_n and $\boldsymbol{\psi}_0 = \left[\phi_0, \phi_1, \ldots, \phi_J\right]^T$ is the initial wavefunction $\phi(x)$ evaluated at $x = x_j$. The matrix $\mathbf{A}$ generalizes to a $(2r+1)$-diagonal matrix of the form

$$\mathbf{A} = \begin{pmatrix} d_0 & a_1 & a_2 & \cdots & a_r & & & & \\ a_1 & d_1 & a_1 & \cdots & a_{r-1} & a_r & & & \\ a_2 & a_1 & d_2 & \cdots & a_{r-2} & a_{r-1} & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & & \\ a_r & a_{r-1} & a_{r-2} & \cdots & d_r & a_1 & & & \\ 0 & a_r & a_{r-1} & \cdots & a_1 & d_{r+1} & & & \\ & & & & & & \ddots & & \\ & & & & & & & d_{J-1} & a_1 \\ & & & & & & & a_1 & d_J \end{pmatrix} \in \mathbb{C}^{J+1\times J+1} \tag{6}$$

where the superscript $a^{(r)}$ is left implicit for compactness.

1.3 Higher-Order Time Evolution

For a higher-order solution in time, we approximate the time advance operator $e^{-iH\Delta t}$ using the Padé approximation

$$e^z = \frac{a_0 + a_1 + \cdots + a_M z^M}{b_0 + b_1 + \cdots + b_M z^M} = \frac{\sum_{m=0}^{M} a_m z^m}{\sum_{m'=0}^{M} b_{m'} z^{m'}} \tag{7}$$

By convention, we set $b_0 = 1$, from which the identity $e^0 = 1$ implies $a_0 = 1$. We solve for a_m and $b_{m'}$ by multiplying Equation 7 by the denominator to get

$$\left(\sum_{m'=0}^{M} b_{m'} z^{m'}\right)\left(\sum_{i=0}^{\infty} c_i z^i\right) = \left(\sum_{m=0}^{M} a_m z^m\right)$$

where we've inserted the Taylor series for e^z. We then multiply the sums on the left side of the equation and equate the coefficients of z to z^{2M}, giving a system of $2M$ equations for $2M$ unknowns a_m and $b_{m'}$. See [3] for a list of the numerator coefficients a_m up to $M = 7$. Once the a_m and $b_{m'}$ are known, it is possible to write the diagonal approximation of the exponential function in the form

$$e^z = \prod_{s=1}^{M}\left(\frac{1 - z/z_s^{(M)}}{1 + z/\bar{z}_s^{(M)}}\right)$$

where $z_s^{(M)}$ are the roots of the numerator in Equation 7, found with the known coefficients a_m. The terms $\bar{z}_s^{(M)}$ are the complex conjugates of $z_s^{(M)}$, although only $z_s^{(M)}$ are needed for this problem. With the z_s^M known, we define the operator $K_s^{(M)}$,

$$K_s^{(M)} \equiv \frac{1 - \frac{iH\Delta t/\hbar}{z_s^{(M)}}}{1 + \frac{iH\Delta t/\hbar}{\bar{z}_s^{(M)}}}$$

which, using the Padé approximant of e^z, allows use to write the time evolution operator as

$$e^{-iH\Delta t/\hbar} \approx \prod_{s=1}^{M} K_s^{(M)}$$

The higher-order time evolution of $\boldsymbol{\psi}$ from $\boldsymbol{\psi}_n$ to $\boldsymbol{\psi}_{n+1}$ now takes $s = 1, 2, \ldots, M$ intermediate steps of the form

$$\boldsymbol{\psi}_{n+1} = e^{-iH\Delta t/\hbar}\boldsymbol{\psi}_n = \prod_{s=1}^{M} K_s^{(M)}\boldsymbol{\psi}_n \tag{8}$$

If we define the intermediate wavefunction

$$\boldsymbol{\psi}_{n+\frac{s}{M}} \equiv K_s^{(M)}\boldsymbol{\psi}_{n+\frac{s-1}{M}},$$

we can solve recursively for $\boldsymbol{\psi}_{n+1}$, starting with

$$\boldsymbol{\psi}_{n+\frac{1}{M}} = K_1^{(M)}\boldsymbol{\psi}_n$$

2 Initial Solution Steps

I must confess that I skipped ahead when solving the problem—I neglected an explicit implementation of the basic $r = 1$, $M = 1$ Crank-Nicolson (CN) solution in Subsection 1.1 and jumped directly to a generalized implementation that could solve for arbitrary r and M, as long as the coefficients $c_k^{(r)}$ and $z_s^{(M)}$ are known. I could then retrospectively investigate the basic CN solution by setting $r = M = 1$. I proceeded as outlined in the following sections.

2.1 Finding Complex Roots $z_s^{(M)}$

I found the complex roots $z_z^{(M)}$ needed for higher-order approximation of the time derivative by constructing a look-up table as shown in the Python code below. The polynomial coefficients, denoted by `c` in the code below, are taken from [3], and the corresponding roots are found with Numpy's `np.roots` function.

```python
import numpy as np
def get_zsM_example(s, M):
    """ Returns the s-th complex roots of the numerator of the M-th order
         diagonal Pade approximation to the exponential function """
    if M == 1:
        c = [1, 2]  # c is the coefficients of the numerator polynomial
    elif M == 2:
        c = [1, 6, 12]  # x^2 + 6x + 12
    # and so on for higher M...
    z_M = np.sort(np.roots(c))  # length-M vector
    return z_M[s-1]  # return s-th element z_sM (using 1-based indexing)
```

I implemented the table up to $M = 8$, but only show up to $M = 2$ for conciseness.

2.2 Finding Coefficients $c_k^{(r)}$

Fornberg [2] gives a powerful algorithm for the coefficients of an arbitrary finite difference scheme, which I implemented in Python to find $c_k^{(r)}$ for arbitrary r.
In our case, $c_k^{(r)}$ are the coefficients of a second-order finite difference approximation containing $2r + 1$ points centered at the middle point. For example, in the basic $r = 1$ finite difference scheme, the coefficients are $1, -2, 1$. Since the coefficients are symmetric about the central point, we only consider the $c_k^{(r)}$ for non-negative k. Table 1 shows the $c_k^{(r)}$ up to $r = 7$. The code for finding the $c_k^{(r)}$ is long, so I'm leaving it out for conciseness. See the functions `get_cr(r)` and `get_ckr(k, r)` in the accompanying `cn.py` file for the implementation.

2.3 Constructing the Matrix A

To find the matrix $\mathbf{A}$ in Equation 6, I first implemented functions to find the auxiliary quantity b, the off-diagonal elements $a_k^{(r)}$, and the diagonal elements d_j. These functions are straightforward implementations of Equations 3 and 4 using the known $c_k^{(r)}$ and $z_s^{(M)}$ from the previous step—I'm leaving the code out for brevity.
Using the known b, $a_k^{(r)}$, and d_j, I constructed $\mathbf{A}$ according to Equation 6:

```python
def get_A(x0, dx, dt, J, r, s, M, V, Vargs):
    """ Returns (J+1) by (J+1) complex matrix A for use with a CN scheme """
    d = get_d(x0, dx, dt, J, r, s, M, V, Vargs)  # returns d_j
    A = np.diag(d, k=0)  # initial diagonal
    for k in range(1, r+1, 1):  # off diagonals for k = 1,..., r
        a = np.zeros(J+1-k, dtype=complex)  # preallocate
        a[0:J+1-k] = get_a(dt, dx, k, r, s, M)  # returns a_kr
        A += np.diag(a, k=k) + np.diag(a, k=-k)  # add off-diagonal
    return A
```

I've left out docstring documentation for conciseness—the notation for `x0` through `M` is the same as used in the rest of the report; `V` and `Vargs` are the potential energy function and its extra parameters (e.g. k for the harmonic potential).

2.4 Solving for the Wavefunction

To keep this section somewhat concise, I assume the reader is familiar with the contents of Subsections 1.1 through 1.3 (or the equivalent content in van Dijk [1]).

With the matrix $\mathbf{A}$ known, I solved for the wavefunction $\psi(x,t)$ by solving the system of equations

$$\mathbf{A}\boldsymbol{\psi}_{n+1} = \mathbf{A}^*\boldsymbol{\psi}_n$$

(as in Eq. 5). For a single time step, i.e. $M = 1$ we have simply

$$\boldsymbol{\psi}_{n+1} = \mathbf{A}^{-1}\mathbf{A}^*\boldsymbol{\psi}_n \implies \boldsymbol{\psi}_n = \left(\mathbf{A}^{-1}\mathbf{A}^*\right)^n\boldsymbol{\phi}$$

Alternatively, for $n = 1, 2, \ldots, N$, repeatedly solve the system

$$\mathbf{A}\boldsymbol{\psi}_{n+1} = \mathbf{A}^*\boldsymbol{\psi}_n$$

using a banded matrix solving scheme, e.g. SciPy's `solve_banded`. In this report I used the simpler multiplication by the inverse $\mathbf{A}^{-1}$ given above.

For multiple time steps $M = 2, 3, \ldots$ we have (see Eq. 8)

$$\boldsymbol{\psi}_{n+1} = \left(\mathbf{A}_M^{-1}\mathbf{A}_M^*\right)\left(\mathbf{A}_{M-1}^{-1}\mathbf{A}_{M-1}^*\right)\cdots\left(\mathbf{A}_1^{-1}\mathbf{A}_1^*\right)\boldsymbol{\psi}_n = \prod_{s=1}^{M}\left(\mathbf{A}_s^{-1}\mathbf{A}_s^*\right)\boldsymbol{\psi}_n$$

where $\mathbf{A}_s$ is analogous to $\mathbf{A}$ as defined in Equation 6, but with $z_1^{(1)}$ replaced by $z_s^{(M)}$. The matrices $\left(\mathbf{A}_s^{-1}\mathbf{A}_s^*\right)$ commute and can be applied in any order. Conveniently, the product of the $\left(\mathbf{A}_s^{-1}\mathbf{A}_s^*\right)$ can be computed once and reused for all n. This leads to the expression

$$\boldsymbol{\psi}_n = \left[\prod_{s=1}^{M}\left(\mathbf{A}_s^{-1}\mathbf{A}_s^*\right)\right]^n\boldsymbol{\phi}$$

where $\boldsymbol{\phi}$ holds the initial wavefunction $\phi(x)$.

3 The Problem's Relevant Quantum Mechanics

3.1 Particle in a Harmonic Potential

For the harmonic potential, the Schrödinger equation in natural units reads

$$\left(i\frac{\partial}{\partial t} + \frac{1}{2}\frac{\partial^2}{\partial x^2} - \frac{1}{2}kx^2\right)\psi(x,t) = 0$$

I use the initial wavefunction

$$\psi(x,0) = \frac{\alpha^{1/2}}{\pi^{1/4}}e^{-\frac{\alpha^2}{2}(x-a)^2}$$

where $\alpha^4 = \frac{mk}{\hbar^2} \to k$, $\omega = \sqrt{\frac{k}{m}} \to \sqrt{k}$ and a is the initial displacement from the origin. The analytic solution for this problem is the coherent state

$$\psi(x,t) = \frac{\alpha^{1/2}}{\pi^{1/4}} \exp\left[-\frac{1}{2}\left(\xi - \xi_0 \cos\omega t\right)^2 - i\left(\frac{\omega t}{2} + \xi\xi_0 \sin\omega t - \frac{1}{4}\xi_0^2 \sin 2\omega t\right)\right]$$

where $\xi = \alpha x$ and $\xi_0 = \alpha a$. Use $\omega = 0.2$, $a = 10$ and $x \in [x_0, x_J] = [-40, 40]$. The oscillatory period is thus $T = \frac{2\pi}{\omega} = 10\pi$.

3.2 Free Gaussian Wave Packet

The Schrödinger equation in natural units for the free particle reads

$$\left(i\frac{\partial}{\partial t} + \frac{1}{2}\frac{\partial^2}{\partial x^2}\right)\psi(x,t) = 0$$

I use the initial wavefunction

$$\psi(x,0) = (2\pi\sigma_0^2)^{-1/4} e^{ik_0(x-a)} \exp\left(-\frac{(x-a)^2}{(2\sigma_0)^2}\right)$$

The analytic solution is

$$\psi(x,t) = \frac{(2\pi\sigma_0^2)^{-1/4}}{\sqrt{1 + \frac{it}{2\sigma_0^2}}} \exp\left[\frac{-\frac{(x-a)^2}{(2\sigma_0)^2} + ik_0(x-a) - \frac{i}{2}k_0^2 t}{1 + \frac{it}{2\sigma_0^2}}\right]$$

I use the parameters $\sigma_0 = \frac{1}{20}$, $k_0 = 50\pi$ and $a = 0.25$. I used the position grid $x \in [x_0, x_J]$ with $x_0 = -0.5$ and x_J varying from 1.5 to 4.5.

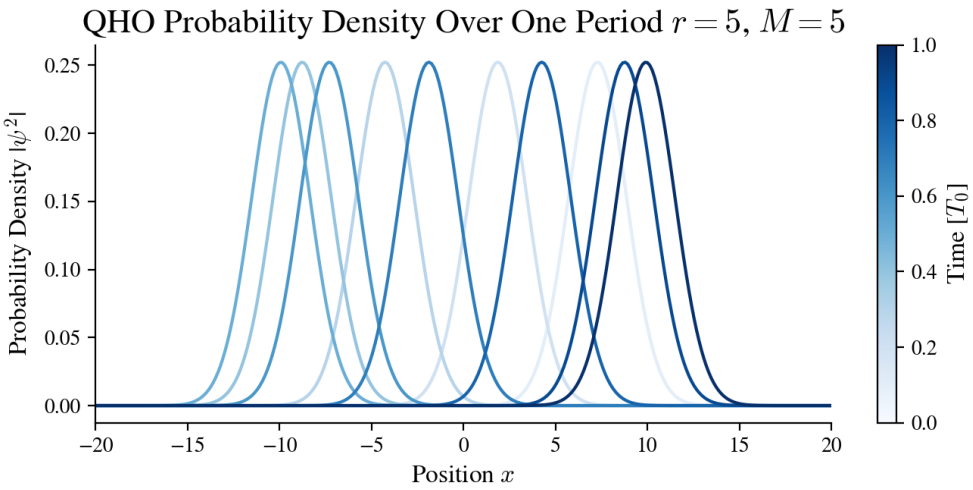

Figure 2: The coherent state's probability density over one period T_0—the color gradient shows progression through time. Found with $r = M = 5$ and $\Delta t = 0.1\pi$.

4 Coherent State in a Harmonic Potential

After a mountain of theory, I will try to let the graphics do the talking for the remaining sections. Figure 2 (above) shows the quantum harmonic oscillator's (QHO) coherent state probability density over one period. As expected, the probability oscillates with amplitude $a = 10$ about the equilibrium position $x = 0$.

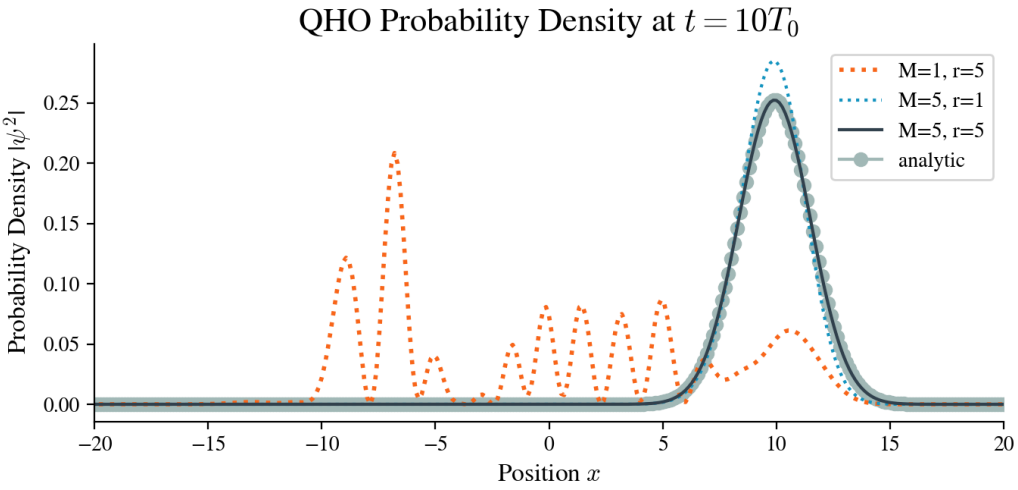

Figure 3: The coherent state's probability density after ten-period simulation for various r and M with $\Delta t = 0.2\pi$. The analytic solution is plotted with gray circles for reference. Note that the first-order-in-time $M = 1$ solution fails completely.

Figure 3 shows the QHO's coherent state after a ten-period simulation time for various r and M using a time step $\Delta t = 0.2\pi$. The $r = M = 5$ solution show excellent agreement with the analytic solution; the $M = 5, r = 1$ solution roughly preserves the initial shape of a Gaussian wave packet; the $M = 1, r = 5$ solution fails completely. The results suggests that a higher-order time approximation (higher M) improves the solution more than a higher-order position approximation (higher r). Interestingly, I found this trend was reversed for the free wave packet.

4.1 Accuracy

I chose to quantify the accuracy of the numerical solution with the error

$$\mathcal{E} = \int_{x_0}^{x_j} |\psi(x, t_N) - \psi_{\text{analytic}}(x, t_N)|^2 \, \mathrm{d}x \tag{9}$$

where ψ and ψ_{analytic} give the numerical and analytic wavefunction solutions, respectively, at the simulation end time t_N. Figure 4 show a three-dimensional visualization of error $\mathcal{E}$ versus M and r for a ten-period simulation with $t_N = 10T_0$ and a time step $\Delta t = 0.2\pi$. I found the enormous decrease in error (over 10 orders of magnitude) from $r = M = 1$ to the higher-order $r = M = 8$ quite impressive.

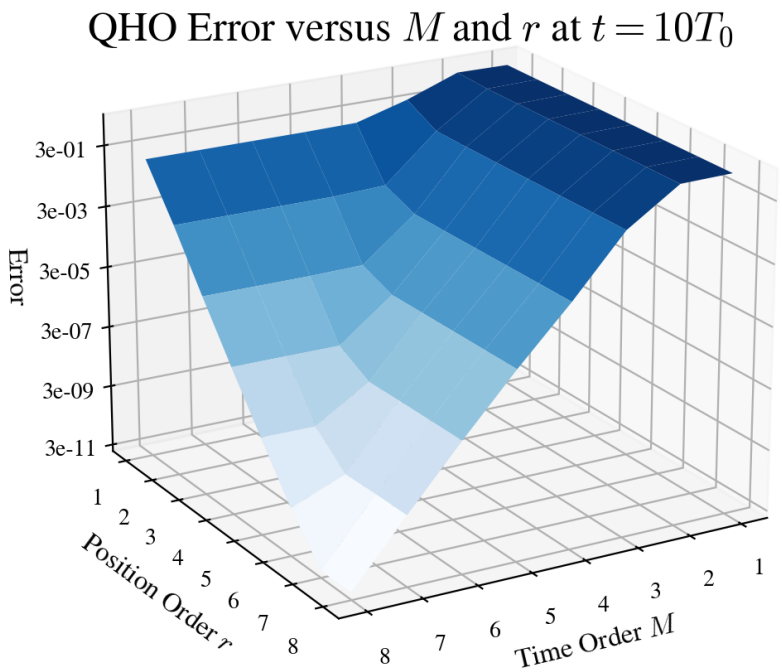

Figure 4: Error in the coherent state solution after a ten-period simulation as a function of r and M—note the logarithmic scale. High M improves the solution more than high r (compare the curves at $r \equiv 1$ and $M \equiv 1$. Found with $\Delta t = 0.2\pi$; error is calculated according to Equation 9.

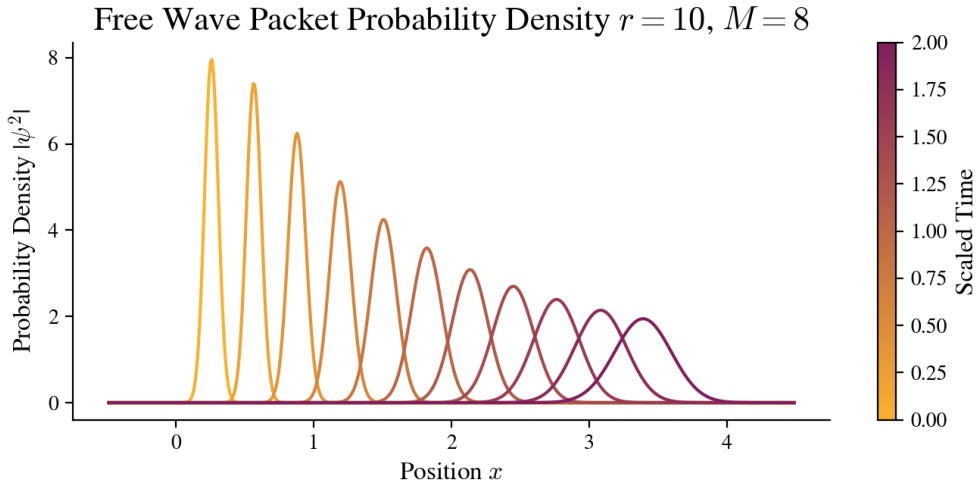

Figure 5: The free wave packet's probability density traveling along the x axis for $[t_0, t_N] = [0, 0.02]$—the color gradient shows progression through time. Found with $r = 10, M = 8$ and $N = 500$.

5 Gaussian Wave Packet in Free Space

Figure 5 (retaken from the introductory Figure 1) shows the free wave packet's probability density as the packet travels from $a = 0.25$ to $a \approx 3.5$, corresponding to

$[t_0, t_N] = [0, 0.02]$ with $N = 500$. I found such a large range in x is only possible with higher-order methods—assuming a reasonable N and J, the basic $r = M = 1$ CN method produces incorrect results for $x \gtrsim 1$. As expected from the theory, the wave packet slowly spreads out as it travels along the x axis.

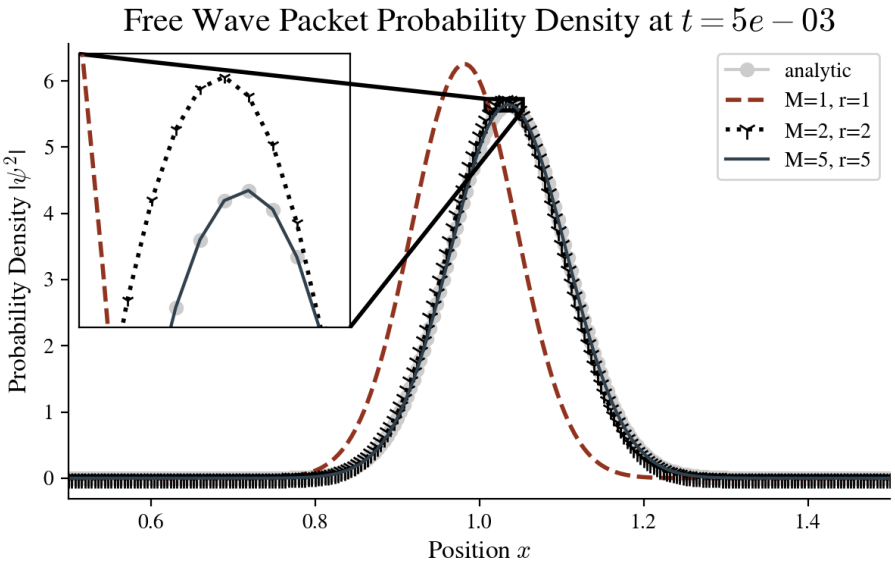

Figure 6: The free wave packet's probability density at $x \approx 1$ for various r and M, using $N = 500$. The analytic solution is plotted with gray circles for reference. Although the $r = M = 2$ and $r = M = 5$ solutions appear macroscopically similar, the zoomed inset shows $r = M = 5$ is superior.

Figure 6 show the free wave packet's probability density at $x \approx 1$ for various r and M using $N = 500$. The free wave packet is less demanding than the QHO—even the $r = M = 1$ solution preserves the wave packet's general shape, while the $r = M = 2$ solution shows good agreement with the analytic solution, although a closer look at the zoomed inset reveals the $r = M = 5$ solution is decidedly superior.

5.1 Accuracy

I quantified the numerical solution's accuracy using the same definition of error (Eq. 9) as for the QHO coherent state. Figure 7 show a three-dimensional visualization of error versus M and r after the wave packet's journey from $a = 0.25$ to $a \approx 1.0$ (corresponding to times in the range $[t_0, t_N] = [0, 5 \cdot 10^{-3}]$) using $N = 500$. As before, the enormous decrease in error with increasing approximation order is quite impressive. Interestingly, the results in Figure 7 show that for the free wave packet, a better approximation of the position derivative (large r) improves the solution more than a higher-order approximation in time (higher M), which is the opposite of the trend seen with the QHO coherent state.

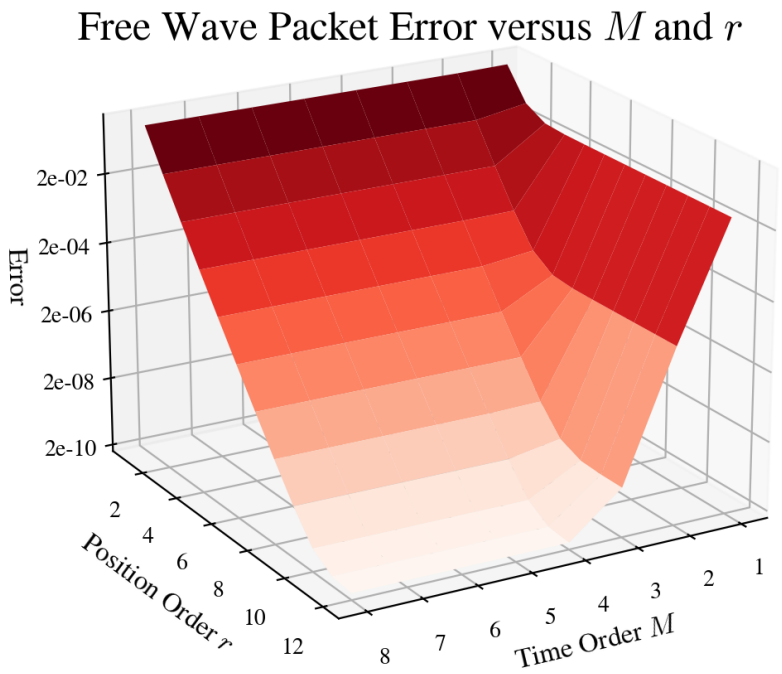

Figure 7: Error in the numerical solution for the wave packet at $a \approx 1.0$ as a function of r and M—note the logarithmic scale. Higher r improves the solution more than high M (compare the curves at $r \equiv 1$ and $M \equiv 1$. Found with $N = 500$.

5.2 A Note on Efficiency and Computation Time

A complete analysis of the Crank-Nicolson method would consider the computation time needed to find a solution of a given accuracy for various values of r, M, J and N. I ran out of time to present such a study in my report, although I certainly experimented with efficiency informally. I found that increasing r and M lowered computation dramatically when searching for a solution at a given error. This improvement in efficiency, like the improvement in accuracy in Figures 4 and 7, spanned many orders of magnitude. For lack of an original option, I would direct the reader to Tables III and IV on pages 7 and 8, respectively, of [1] for a formal study of efficiency.

References

[1] W. van Dijk and F. M. Toyama. "Accurate numerical solutions of the time-dependent Schrödinger equation." Phys. Rev. E **75**, 036707 (2007). https://arxiv.org/pdf/physics/0701150.pdf

[2] Bengt Fornberg, "Generation of finite difference formulas on arbitrarily spaced grids", In Mathematics of Computation **51**, 184, pp. 699-706, 1988. https://web.njit.edu/~jiang/math712/fornberg.pdf

[3] Barry, Paul. "A119274 Triangle of coefficients of numerators in Padé approximation to exp(x)". *The On-Line Encyclopedia of Integer Sequences*. 12 May 2006. https://oeis.org/A119274