Introduction:

The Schrodinger equation is a linear partial differential equation that governs the wave function of a quantum-mechanical system. We solve for the wave function, ψ(x,t), and some other values such as the real part, imaginary part, and modulus of ψ. For this project, we are solving the one-dimensional and two-dimensional time-dependent Schrodinger equations using the Crank-Nicolson method and the Alternating Direction Implicit (ADI) method for each problem respectively.

Review of Theory and Numerical Approach:

For the first problem, we are solving the 1D time-dependent Schrodinger equation which takes the form of the continuum equation:

$$i\psi(x,t)_t = -\psi_{xx} + V(x,t)\psi$$

Where ψ(x,t) is some complex wave function and the equation is solved on the domain of $0 \le x \le 1$ and $0 \le t \le t_{max}$ with initial and boundary conditions of:

$$\psi(x,0) = \psi_0(x)$$
$$\psi(0,t) = \psi(1,t) = 0$$

This wave function also has a real component, Re(ψ) and imaginary component, Im(ψ), as well as some modulus |ψ| and a probability density defined by:

$$P(x,t) = \int_0^x \psi(\bar{x},t)\psi^*(\bar{x},t)d\bar{x}$$

P(x,t) uses the trapezoidal formula to solve the integral.

The exact family of solutions for this Schrodinger equation is:

$$\psi(x,t) = e^{-im^2\pi^2 t} sin(m\pi x)$$

To solve this 1D Schrodinger equation, we used the Crank-Nicolson discretization approach, a type of finite-difference method that is used to solve partial differential equations such as this one.

To set up this discretization, we must first create a mesh defined by first taking the ratio of temporal to spatial mesh spacings $\lambda = \Delta t/\Delta x$ and some discretization level, ℓ, to get the following:

$$n_x = 2^l + 1$$
$$\Delta x = 2^{-l}$$
$$\Delta t = \lambda\Delta x$$
$$n_t = round(t_{max}/\Delta t) + 1$$

We then move onto get the Crank-Nicolson scheme as follows:

$$i\frac{\psi_j^{n+1}+\psi_j^n}{\Delta t} = -\frac{1}{2}\left(\frac{\psi_{j+1}^{n+1}-2\psi_j^{n+1}+\psi_{j-1}^{n+1}}{\Delta x^2} + \frac{\psi_{j+1}^{n}-2\psi_j^{n}+\psi_{j-1}^{n}}{\Delta x^2}\right) + \frac{1}{2}V_j^{n+\frac{1}{2}}(\psi_j^{n+1} + \psi_j^n)$$

Where j = 2, 3, …, nx-1 and n = 1, 2, …, nt-1

This scheme gets divided into a sparse matrix **A**, comprising three parts, ψ_{j+1}^{n+1}, ψ_j^{n+1}, ψ_{j-1}^{n+1} that make up a tridiagonal system. Each of these has some coefficient (dl, d, du) defined by:

$$dl = \frac{1}{2\Delta x^2}$$
$$d = \frac{i}{\Delta t} - \frac{1}{\Delta x^2} - \frac{V_j^{n+1/2}}{2}$$
$$du = dl$$

This matrix is equal to some f_j^n which also consists of three parts, ψ_{j+1}^n, ψ_j^n, ψ_{j-1}^n similar to **A**. Similarly the coefficients here are:

$$dl = \frac{-1}{2\Delta x^2}$$
$$d = \frac{i}{\Delta t} + \frac{1}{\Delta x^2} + \frac{V_j^{n+1/2}}{2}$$
$$du = dl$$

We then solve the equation:

$$\overline{A}\psi^{n+1} = f^n$$

There are two idtypes:

Exact family (idtype = 0)

$$\psi(x, 0) = sin(m\pi x)$$

Boosted Gaussian (idtype = 1)

$$\psi(x, 0) = e^{ipx} e^{-((x-x_0)/\delta)^2}$$

There are also two vtypes:

No potential (vtype = 0)

$$V(x) = 0$$

Rectangular barrier or well (vtype = 1)

$$V(x) = 0 \; for \; x < x_{min}, V_c \; for \; x_{min} \le x \le x_{max}, \; 0 \; for \; x > x_{max}$$

The second problem has us solving a 2D version of the time-dependent Schrodinger equation which is defined by the continuum equation:

$$i\psi(x, y, t)_t = -(\psi_{xx} + \psi_{yy}) + V(x, y)\psi$$

Where ψ(x,t) is some complex wave function and the equation is solved on the domain of $0 \le x \le 1$, $0 \le y \le 1$, and $0 \le t \le t_{max}$ with initial and boundary conditions of:

$$\psi(x, y, 0) = \psi_0(x, y)$$

$$\psi(0, y, t) = \psi(1, y, t) = \psi(x, 0, t) = \psi(x, 1, t) = 0$$

The exact family of solutions for this Schrodinger equation is:

$$\psi(x, y, t) = e^{-i(m_x^2+m_y^2)\pi^2 t} sin(m_x \pi x) sin(m_y \pi y)$$

To solve this 1D Schrodinger equation, we used the ADI discretization approach.

To set up this discretization, we must first create a mesh defined by first taking the ratio of temporal to spatial mesh spacings $\lambda = \Delta t/\Delta x = \Delta t/\Delta y$ and some discretization level, ℓ, to get the following:

$$n_x = n_y = 2^l + 1$$
$$\Delta x = \Delta y = 2^{-l}$$
$$\Delta t = \lambda \Delta x$$
$$n_t = round(t_{max}/\Delta t) + 1$$

Here instead of the Crank-Nicolson scheme, we set up an ADI scheme:

$$(1 - i\frac{\Delta t}{2}\partial^h_{xx})\psi^{n+1/2}_{i,j} = (1 + i\frac{\Delta t}{2}\partial^h_{xx})(1 + i\frac{\Delta t}{2}\partial^h_{yy} - i\frac{\Delta t}{2}V_{i,j})\psi^n_{i,j}$$

The partials in this equation are defined as:

$$\partial^h_{xx}\psi^n_{i,j} \equiv \frac{\psi^n_{i+1,j} - 2\psi^n_{i,j} + \psi^n_{i-1,j}}{\Delta x^2}$$

$$\partial^h_{yy}\psi^n_{i,j} \equiv \frac{\psi^n_{i,j+1} - 2\psi^n_{i,j} + \psi^n_{i,j-1}}{\Delta x^2}$$

Then this initial equation is used to solve for $\psi^{n+1}_{i,j}$.

$$(1 - i\frac{\Delta t}{2}\partial^h_{yy} + i\frac{\Delta t}{2}V_{i,j})\psi^{n+1}_{i,j} = \psi^{n+1/2}_{i,j},$$
$$i = 2, 3,..., n_x - 1,\ j = 2, 3,..., n_y - 1, n = 1, 2,..., n_t - 1$$

Similarly to in Problem 1, this scheme gets divided into a sparse matrix **D_base**, comprising three parts, $\psi^{n+1/2}_{i+1,j}$, $\psi^{n+1/2}_{i,j}$, $\psi^{n+1/2}_{i-1,j}$ that make up a tridiagonal system. Each of these has some coefficient (dl, d, du) which are gotten by taking the base tridiagonal system and adding or subtracting it from a matrix of ones.

$$dl = -\frac{i\Delta t}{2\Delta x^2}$$
$$d = 1 + \frac{i\Delta t}{\Delta x^2}$$
$$du = dl$$

This sparse matrix, **D_base**, which is now divided into the right and left hand side are used to solve the system giving us $\psi^{n+1/2}$ and eventually ψ^{n+1}.

There are two idtypes:

Exact family (idtype = 0)

$$\psi(x, y, 0) = sin(m_x \pi x)sin(m_y \pi y)$$

Boosted Gaussian (idtype = 1)

$$\psi(x, y, 0) = e^{ip_x x} e^{ip_y y} e^{-((x-x_0)/\delta_x^2)+((y-y_0)/\delta_y^2)}$$

There are also two vtypes:

No potential (vtype = 0)

$$V(x, y) = 0$$

Rectangular barrier or well (vtype = 1)

$$V(x) = V_c \; for \; x_{min} \le x \le x_{max} \; and \; y_{min} \le y \le y_{max}, \; 0 \; otherwise$$

Double slit (vtype = 2)

Let some $j' = (n_y - 1)/4 + 1$
Then:

$$V_{i,j'} = V_{i,j'+1} = V_c \; when \; x_i < x_1 \; or \; x_i > x_4 \; or \; x_2 < x_i < x_3, \; 0 \; otherwise$$

Results:

Problem 1

Convergence Tests

1. Idtype = 0, vtype = 0, idpar = 3, tmax = 0.25, lambda = 0.1, lmin = 6, lmax = 9

We performed two types of convergence for this scenario. The first just took the rms value of $d\psi^{l} = \psi^{l+1} - \psi^{l}$ for four values of ℓ (6, 7, 8, and 9).

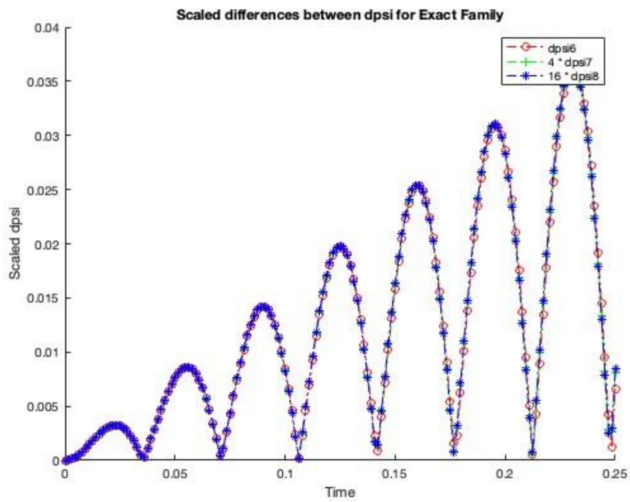

Figure 1: Scaled differences between dpsi (Exact family).

The second type of convergence testing took the rms value of $E(\psi^{l}) = \psi_{exact} - \psi^{l}$ for the same four values of ℓ.

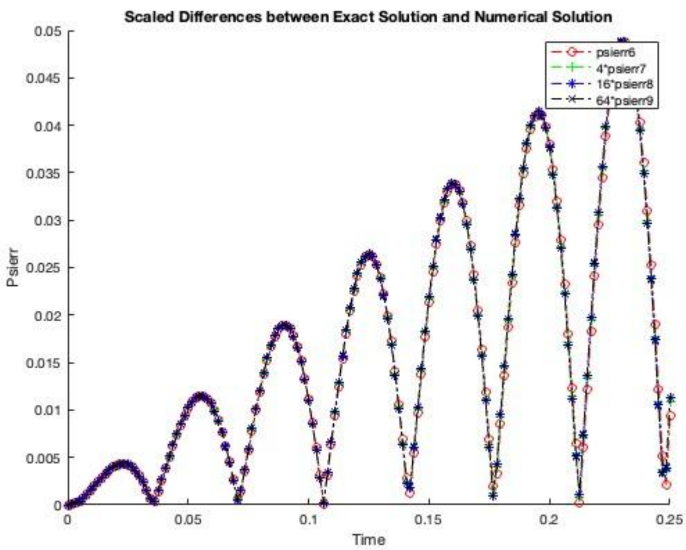

Figure 2: Scaled differences between exact solution and numerical.

2. Idtype = 1, vtype = 0, idpar = [0.50 0.057 0.0], tmax = 0.01, lambda = 0.01, lmin = 6, lmax = 9

For this scenario, we performed just the first type of convergence test.

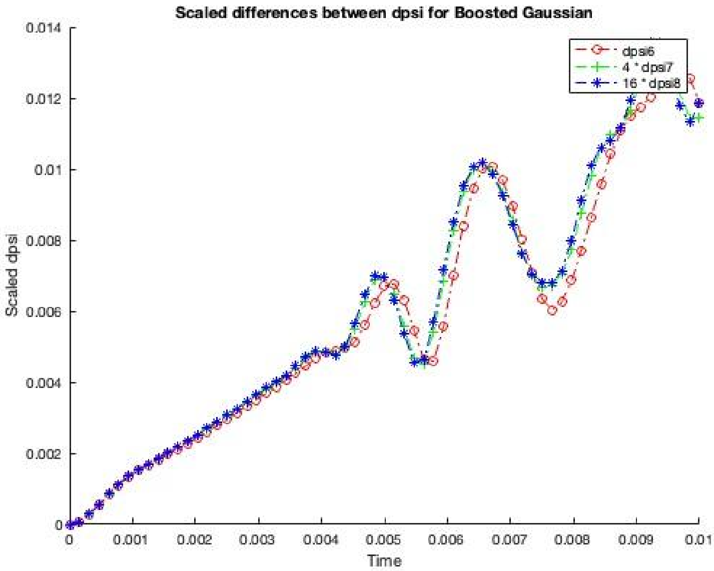

Figure 3: Scaled differences between dpsi (Boosted Gaussian).

All the convergence tests use Re(ψ).

Numerical Experiments

We then performed a couple of numerical experiments that showed a particle left of a barrier of height V0 and another in a well. The two experiments showed the dependence of $ln(F_e(x_1, x_2))$ on $ln(V_0)$. The former was calculated by taking the probability that the particle is within the bounds of x1 and x2 or the fraction of time the particle spends in that interval divided by the difference between the bounds.

$$ln(F_e(x_1, x_2)) \ = \ ln\frac{\overline{P}(x_2) - \overline{P}(x_1)}{x_2 - x_1}$$

1. Tmax = 0.10, level = 9, lambda = 0.01, idtype = 1, idpar = [0.40, 0.075, 20.0], vtype = 1, vpar = [0.8, 1.0, (exp(-2), exp(5), 251)], x1 = 0.8, x2 = 1.0

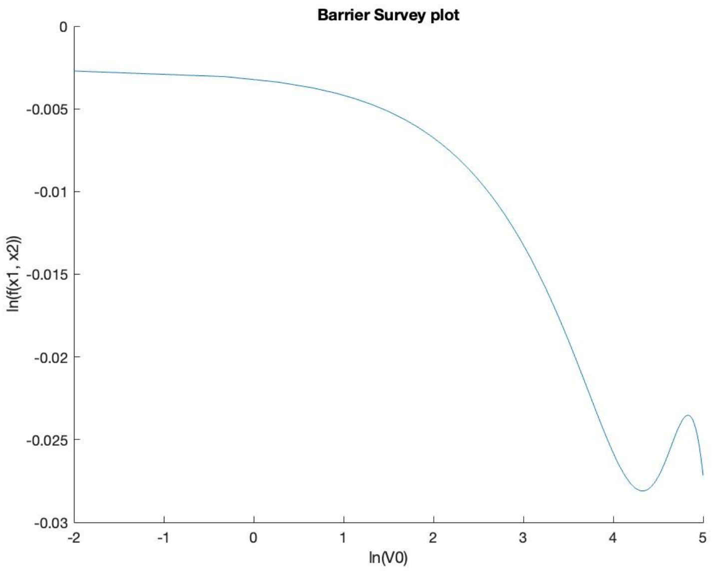

Figure 4: Plot of $ln(F_e(0.8,\ 1.0))\ vs\ ln(V_0)$ for barrier.

This is more or less what we expect as with a higher potential, the wave function has less chance of being inside the barrier. There is a little blip around 4 which probably means there is some error with how the code is calculating probability.

2. Tmax = 0.10, level = 9, lambda = 0.01, idtype = 1, idpar = [0.40, 0.075, 0.0], vtype = 1, vpar = [0.6, 0.8, (-exp(2), -exp(10), 251)], x1 = 0.6, x2 = 0.8

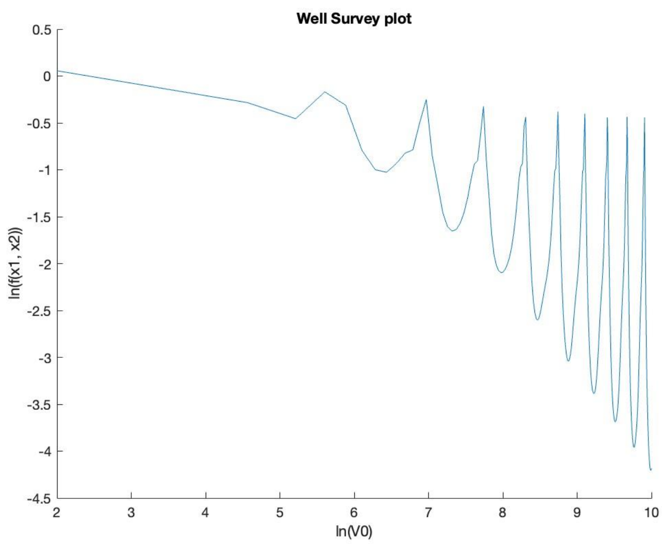

Figure 5: Plot of $ln(F_e(0.6,\ 0.8))\ vs\ ln(V_0)$ for well.

This is also more or less what we expect as we have a decreasing trend with some oscillation. This probably means that the permeability of the wave depends on certain potentials instead of scaling with the magnitude of the potential.

Problem 2

Convergence Test

We performed two types of convergence for this scenario just like for Problem 1.

Idtype = 0, vtype = 0, idpar = [2, 3], tmax = 0.05, lambda = 0.05, lmin = 6, lmax = 9

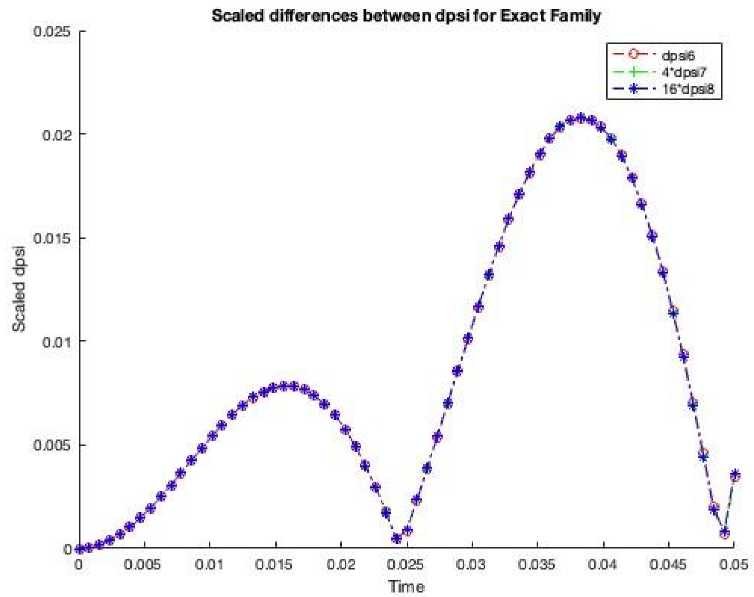

Figure 6: Scaled differences between dpsi (Exact family).

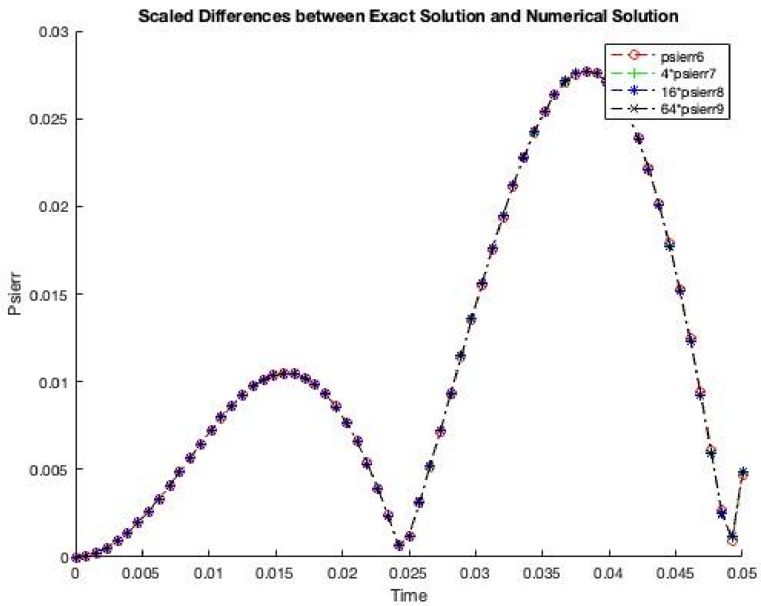

Figure 7: Scaled differences between exact solution and numerical.

Animation

All animations were done using level 6 and with tmax and lambda being 0.05. All animations use **contourf** to show 2D contour of the system.

Idpar = [0.8, 0.8, 0.05, 0.05, -5, -5] & vpar = [0.1, 0.3, 0.1, 0.3, 100000]

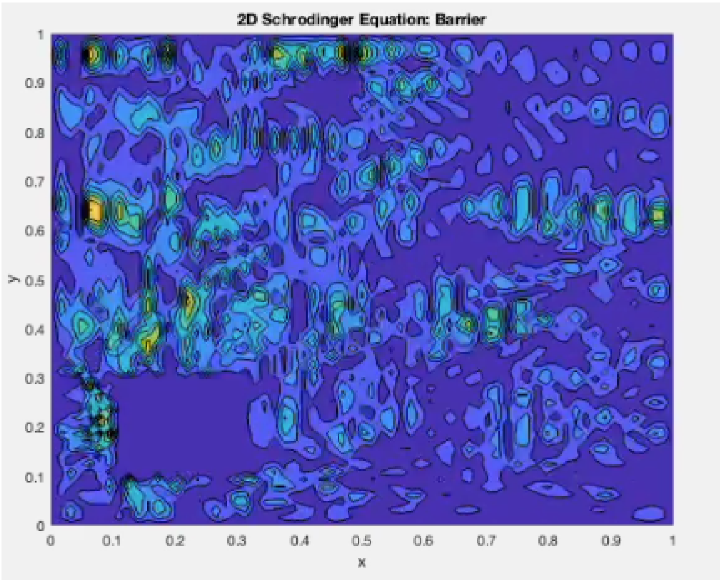

Figure 8: Snapshot of barrier animation.

Idpar = [0.5, 0.5, 0.05, 0.05, -4, -4] & vpar = [0.1, 0.3, 0.1, 0.3, -100000]

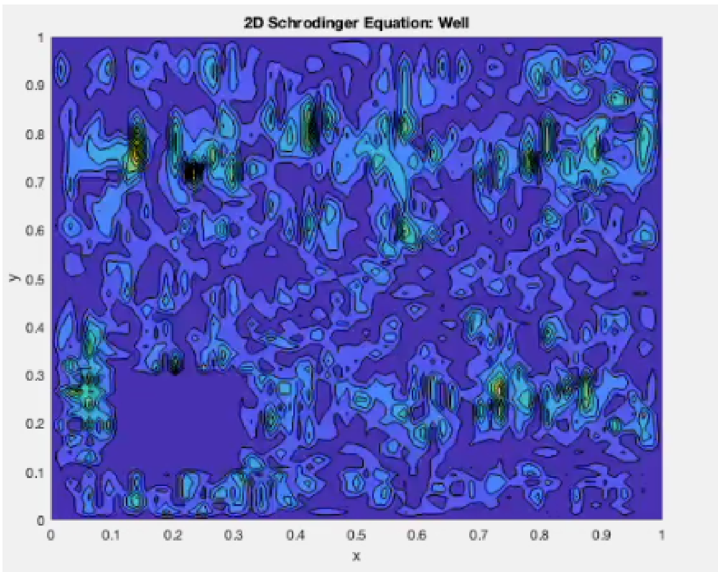

Figure 9: Snapshot of well animation.

Idpar = [0.5, 0.8, 0.15, 0.06, 0, -40] & vpar = [0.25, 0.3, 0.65, 0.7, 100000]

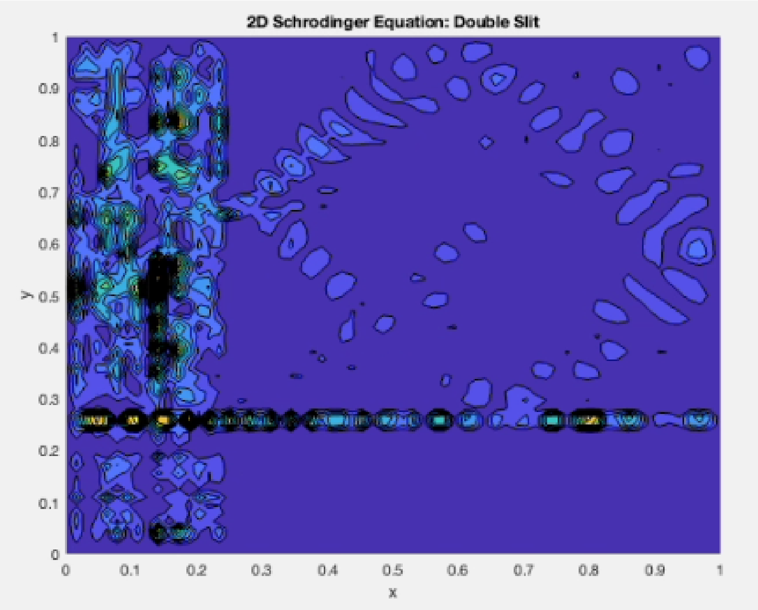

Figure 10: Snapshot of double slit animation.

The plots here don't show too much about the system but all the animations are in the zip file along with all the other MATLAB files and plots.

Discussion/Conclusion:

The big difficulty that consumed the most time was trying to debug the 2d Schrodinger equation. Since the convergence tests took so long and the shapes are 3d (nt, nx, ny) it was hard to get to the root of the problem. It took a lot of command window work and stepping into the function to fix small problems in the code. It was also hard to understand the implementation of the ADI scheme. The convergence test for the 2d Schrodinger equation also takes an extremely long time to run (20 min) for level 9 (takes an extremely short time up until 9) so there could probably be some optimizations for the 2d_adi code.

For this project, you want to run **ctest_1d.m** and **ctest_2d.m** for the convergence tests, **barrier_survey.m** and **well_survey.m** for the numerical experiments as well as **animation.m** for the 2d Schrodinger equation animation.

ChatGPT and other generative AI tools were not used at all for this project.