Math 228B - HW3

Ahmed H. Mahmoud

3 March 2017

1 Problem No.1

1.1 Problem Description:

Consider

$$u_t = 0.1\Delta u \ \ on \ \ \Omega = (0,1)\times(0,1)$$
$$\frac{\partial u}{\partial \vec{n}} = 0 \ \ on \ \ \partial\Omega$$
$$u(x,y,0) = exp(-10((x-0.3)^2+(y-0.4)^2))$$

Write a program to solve this PDE using the Peaceman-Rachford ADI scheme on a cell-centered grid. Use a direct solver for the tridiagonal systems. In a cell-centered discretization the solution is stored at the grid points $(x_i, y_i) = (\Delta x(i-0.5), \Delta x(j-0.5))$ for $i, j = 1...N$ and $dx = 1/N$. This discretization is natural for handling Neumann boundary conditions, and it is often used to discretize conservation laws. At the grid points adjacent to the boundary, the one-dimensional discrete Laplacian for homogeneous Neumann boundary conditions is

$$u_{xx}(x_1) \approx \frac{-u_1+u_2}{\Delta x^2}$$

1. Perform a refinement study to show that your numerical solution is second-order accurate is space and time (refine time and space simultaneously using $\Delta t \propto \Delta x$) at time $t = 1$.

2. Time your code for different grid sizes. Show how the computational time scales with the gird size. Compare your timing results with those from the previous homework assignment for Crank-Nicolson. If you had error in your codes from HW2, fix them.

   Note, many of you are using script languages (MATLAB and python), and loops in these languages can be slow. You can program a single time step of ADI (in 2D) without any loops. For example, you can solve $LU = F$, where U and F are stored as arrays rather than vectors, and L is the 1D operator to invert. Avoiding loops is not necessary for this assignment. Write your code for correctness and clarity first, and efficiency later if you have time.

3. Show that the spatial integral of the solution of the PDE does not change in time. That is

$$\frac{d}{dt}\int_{\Omega} u dV = 0$$

4. Show that the solution to the discrete equations satisfies the discrete conservation property

$$\sum_{i,j} u_{i,j}^n = \sum_{i,j} u_{i,j}^0$$

for all n. Demonstrate this property with your code.

1.2 Solution:

Part 1: The two steps ADI scheme involves decoupling of the spatial discretization such that only one spatial direction discretization (x-sweep) is performed and advances to intermediate time step followed by the opposite spatial direction (y-sweep)and advances to the next time step. The system produces a tri-diagonal system of equation to be solved at each sweep step. The equation can be expressed under this scheme as:

$$\frac{U_{i,j}^{n+\frac{1}{2}} - U_{i,j}^n}{\frac{\Delta t}{2}} = 0.1\left(\frac{U_{i+1,j}^{n+\frac{1}{2}} - 2U_{i,j}^{n+\frac{1}{2}} + U_{i-1,j}^{n+\frac{1}{2}}}{(\Delta x)^2} + \frac{U_{i,j+1}^n - 2U_{i,j}^n + U_{i,j-1}^n}{(\Delta y)^2}\right) \tag{1a}$$

$$\frac{U_{i,j}^{n+1} - U_{i,j}^{n+\frac{1}{2}}}{\frac{\Delta t}{2}} = 0.1\left(\frac{U_{i+1,j}^{n+\frac{1}{2}} - 2U_{i,j}^{n+\frac{1}{2}} + U_{i-1,j}^{n+\frac{1}{2}}}{(\Delta x)^2} + \frac{U_{i,j+1}^{n+1} - 2U_{i,j}^{n+1} + U_{i,j-1}^{n+1}}{(\Delta y)^2}\right) \tag{1b}$$

where equation (1a) represents the x-sweep and (1b) represents the y-sweep. After rearranging the equations, we obtain

$$(1+2r)U_{i,j}^{n+\frac{1}{2}} - rU_{i-1,j}^{n+\frac{1}{2}} - rU_{i+1,j}^{n+\frac{1}{2}} = (1-2r)U_{i,j}^n + rU_{i,j-1}^n + rU_{i,j+1}^n \tag{2a}$$

$$(1+2r)U_{i,j}^{n+1} - rU_{i,j-1}^{n+1} - rU_{i,j+1}^{n+1} = (1-2r)U_{i,j}^{n+\frac{1}{2}} + rU_{i-1,j}^{n+\frac{1}{2}} + rU_{i+1,j}^{n+\frac{1}{2}} \tag{2b}$$

where $r = 0.1\Delta t/2h^2$ and $h = \Delta x = \Delta y$. Equation (2) is used for the internal grid points. For the grid points adjacent to the boundary, we use $u_{xx}(x_1) \approx (-u_1 + u_2)/\Delta x^2$ which can be derived by adding virtual points extends outside the physical domain and have values equal to the (numerical domain) boundary points. The equation can be discretized to

$$\frac{U_{i,j}^{n+\frac{1}{2}} - U_{i,j}^n}{\frac{\Delta t}{2}} = 0.1\left(\frac{U_{i+1,j}^{n+\frac{1}{2}} - U_{i,j}^{n+\frac{1}{2}}}{h^2} + \frac{U_{i,j+1}^n - U_{i,j}^n}{h^2}\right)$$

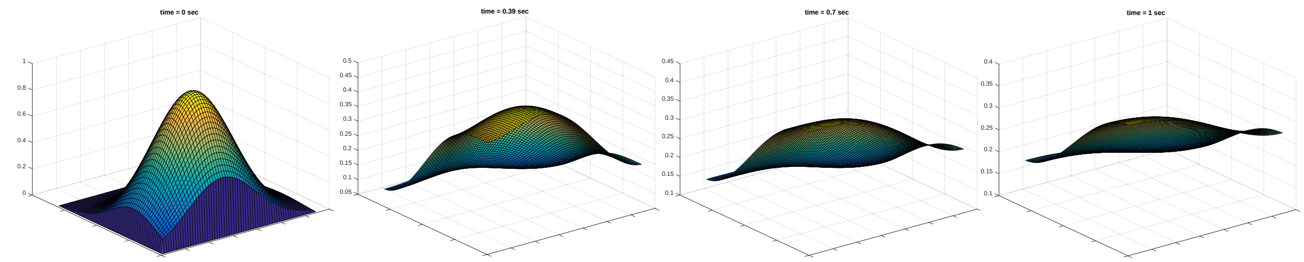
$$\frac{U_{i,j}^{n+1} - U_{i,j}^{n+\frac{1}{2}}}{\frac{\Delta t}{2}} = 0.1\left(\frac{U_{i,j+1}^{n+1} - U_{i,j}^{n+1}}{h^2} + \frac{U_{i+1,j}^{n+\frac{1}{2}} - U_{i,j}^{n+\frac{1}{2}}}{h^2}\right)$$

After rearranging, the equation becomes

$$(1+r)U_{i,j}^{n+\frac{1}{2}} - rU_{i+1,j}^{n+\frac{1}{2}} = (1-r)U_{i,j}^n + rU_{i,j+1}^n \tag{3a}$$

$$(1+r)U_{i,j}^{n+1} - rU_{i,j+1}^{n+1} = (1-r)U_{i,j}^{n+\frac{1}{2}} + rU_{i+1,j}^{n+\frac{1}{2}} \tag{3b}$$

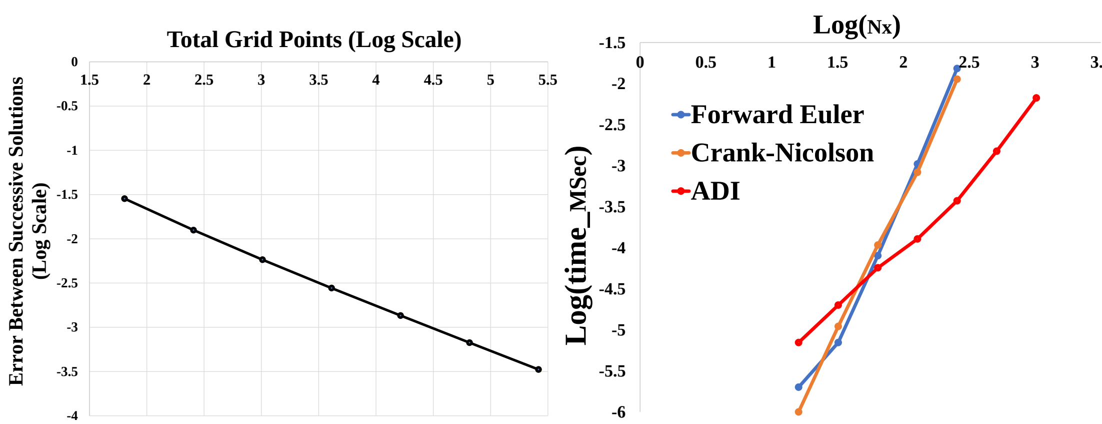
Since we are using cell-centered grid, the numerical solution extends from $h/2$ to $1 - h/2$ in both x and y directions, where the physical domain extends from $[0, 1]$. Thus, the smaller h, the closer the numerical domain gets closer to the physical domain. The expanded form of equations (2) and (3) and matrix formulation can be found in Appendix. The matrix formulation suggests that a tridiagonal system will be solved ny times for the x-sweep and nx times for y-sweep. The solution at different time steps on a grid of size 64×64 is shown in Figure 1.



(a) time=0.0 (b) time=0.39 (c) time=0.7 (d) time=1.0

Figure 1: The solution of problem 1 on a gird of size 64×64 and time step $\Delta t = 0.001562$ as the time advances.

Refinement Study: we used a similar approach as in homework 1 to carry on the refinement study. We start we certain time and space steps and calculate the sum of the solution all over the grid points. Next, we halve the time and space steps and get the sum. We then calculate the error as the absolute difference between the normalized sum of two successive solution. We then plot the total number of gird points Vs. the error on a log-log scale as shown in Figure 2. The slope of the line is ≈ 2.0 which indicates that the scheme used in second order accurate in space and time.



(a) Refinement Study (b) Timing Comparison

Figure 2: Refinement study (left) shows that ADI is second order accurate in time and space. Timing comparison (right) shows the order of computational time scale for ADi, CN and Forward Euler.

Part 2: We performed a comparison between the ADI, Crank-Nicolson and Forward Euler where time spacing is fixed to 0.01. The results are shown in Figure 2 (b) on a log-log scale between Δx

and time in milliseconds. It is found that for small gird size, using Crank-Nicolson is more efficient. But after a certain limit ($\approx 64 \times 64 gridpoints$), ADI becomes more superior. This is due to the fact that ADI solve a tridiagonal system which is faster to solve than the banded matrix solver for Crank-Nicolson, specially that the traditional matrix solver was optimized to take in constants (not array in C++) since the value of a single diagonal are the same.

Part 3: This part asks to prove that the concentration will not change with time which is implied due to the insulated boundary (following the conservation of mass), or $(d/dt)\int_\Omega u.dV = 0$. We can change the order of differentiation and integration and since $u_t = 0.1\Delta u$, then we get

$$\frac{d}{dt}\int_\Omega u.dV = \int_\Omega \frac{du}{dt}.dV = \int_\Omega 0.1\Delta u.dV$$

Since u is function in x and y and each extends from 0 to 1:

$$\frac{d}{dt}\int_\Omega u.dV = 0.1\int_0^1\int_0^1 \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}.dxdy = 0.1\Big(\int_0^1\int_0^1 \frac{\partial^2 u}{\partial x^2}.dxdy + \int_0^1\int_0^1 \frac{\partial^2 u}{\partial y^2}.dydx\Big)$$

Using the fundamental theorem of calculus to obtain solution for single integration, we get

$$\int_0^1 \frac{\partial^2 u}{\partial x^2}.dx = \frac{\partial u^2}{\partial x^2}|_{x=1} - \frac{\partial u^2}{\partial x^2}|_{x=0}$$

The boundary conditions suggest that there no material crossing the boundary and thus $\partial u/\partial x = 0$ along the boundary ($x = 0$ and $x = 1$). Thus

$$\int_0^1 \frac{\partial^2 u}{\partial x^2}.dx = 0$$

Similarly, we can prove that

$$\int_0^1 \frac{\partial^2 u}{\partial y^2}.dy = 0$$

Plugging these results back to the original equation, we get

$$\frac{d}{dt}\int_\Omega u.dV = 0.1\int_0^1\int_0^1 \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}.dxdy = 0$$

Part 4: For this part, if we prove that the summation equals for two time steps $t = n$ and $t = n+1$, then it is proven for the time steps $t = 0$ and $t = n$ (by the virtue that the summation for all intermediate time steps between $t = 0$ and $t = n$ are the same). We start by writing the first couple of equations for the x-sweep which will give us the summation $\sum_{i,j} U^n_{i,j}$. Here we use small and capital u interchangeable.

$$i=0,j=0 \rightarrow (1+r)U_{0,0}^{n+\frac{1}{2}} - rU_{1,0}^{n+\frac{1}{2}} = (1-r)U_{0,0}^{n} + rU_{0,1}^{n}$$

$$i=1,j=0 \rightarrow (1+2r)U_{1,0}^{n+\frac{1}{2}} - rU_{0,0}^{n+\frac{1}{2}} - rU_{2,0}^{n+\frac{1}{2}} = (1-r)U_{1,0}^{n} + rU_{1,1}^{n}$$

$$i=2,j=0 \rightarrow (1+2r)U_{2,0}^{n+\frac{1}{2}} - rU_{1,0}^{n+\frac{1}{2}} - rU_{3,0}^{n+\frac{1}{2}} = (1-r)U_{2,0}^{n} + rU_{2,1}^{n}$$

....
....

$$i=0,j=1 \rightarrow (1+r)U_{0,1}^{n+\frac{1}{2}} - rU_{1,1}^{n+\frac{1}{2}} = (1-2r)U_{0,1}^{n} + rU_{0,0}^{n} + rU_{0,2}^{n}$$

$$i=1,j=1 \rightarrow (1+2r)U_{1,1}^{n+\frac{1}{2}} - rU_{0,1}^{n+\frac{1}{2}} - rU_{2,1}^{n+\frac{1}{2}} = (1-2r)U_{1,1}^{n} + rU_{1,0}^{n} + rU_{1,2}^{n}$$

$$i=2,j=1 \rightarrow (1+2r)U_{2,1}^{n+\frac{1}{2}} - rU_{1,1}^{n+\frac{1}{2}} - rU_{3,1}^{n+\frac{1}{2}} = (1-2r)U_{2,1}^{n} + rU_{2,0}^{n} + rU_{2,2}^{n}$$

....
....

$$i=0,j=2 \rightarrow (1+r)U_{0,2}^{n+\frac{1}{2}} - rU_{1,2}^{n+\frac{1}{2}} = (1-2r)U_{0,2}^{n} + rU_{0,1}^{n} + rU_{0,3}^{n}$$

$$i=1,j=2 \rightarrow (1+2r)U_{1,2}^{n+\frac{1}{2}} - rU_{0,2}^{n+\frac{1}{2}} - rU_{2,2}^{n+\frac{1}{2}} = (1-2r)U_{1,2}^{n} + rU_{1,1}^{n} + rU_{1,3}^{n}$$

$$i=2,j=2 \rightarrow (1+2r)U_{2,2}^{n+\frac{1}{2}} - rU_{1,2}^{n+\frac{1}{2}} - rU_{3,2}^{n+\frac{1}{2}} = (1-2r)U_{2,2}^{n} + rU_{2,1}^{n} + rU_{2,3}^{n}$$

....
....

Now we sum, the right hand side will give

$$U_{0,0}^{n} + U_{0,1}^{n} + U_{0,2}^{n} + ...U_{1,0}^{n} + U_{1,1}^{n} + U_{1,2}^{n} + ...U_{2,0}^{n} + U_{2,1}^{n} + U_{2,2}^{n} + ... = \sum_{i,j} U_{i,j}^{n}$$

Summing the left hand side gives

$$U_{0,0}^{n+\frac{1}{2}} + U_{0,1}^{n+\frac{1}{2}} + U_{0,2}^{n+\frac{1}{2}} + ...U_{1,0}^{n+\frac{1}{2}} + U_{1,1}^{n+\frac{1}{2}} + U_{1,2}^{n+\frac{1}{2}} + ...U_{2,0}^{n+\frac{1}{2}} + U_{2,1}^{n+\frac{1}{2}} + U_{2,2}^{n+\frac{1}{2}} + ...$$
$$= \sum_{i,j} U_{i,j}^{n+\frac{1}{2}}$$

We can do the same process for the y-sweep to obtain $\sum_{i,j} U_{i,j}^{n+1}$ such that

$$i = 0, j = 0 \rightarrow (1+r)U_{0,0}^{n+1} - rU_{0,1}^{n+1} = (1-r)U_{0,0}^{n+\frac{1}{2}} + rU_{1,0}^{n+\frac{1}{2}}$$

$$i = 0, j = 1 \rightarrow (1+2r)U_{0,1}^{n+1} - rU_{0,0}^{n+1} - rU_{0,2}^{n+1} = (1-r)U_{0,1}^{n+\frac{1}{2}} + rU_{1,1}^{n+\frac{1}{2}}$$

$$i = 0, j = 2 \rightarrow (1+2r)U_{0,2}^{n+1} - rU_{0,1}^{n+1} - rU_{0,3}^{n+1} = (1-r)U_{0,2}^{n+\frac{1}{2}} + rU_{1,2}^{n+\frac{1}{2}}$$

....

....

$$i = 1, j = 0 \rightarrow (1+r)U_{1,0}^{n+1} - rU_{1,1}^{n+1} = (1-2r)U_{1,0}^{n+\frac{1}{2}} + rU_{0,0}^{n+\frac{1}{2}} + rU_{2,0}^{n+\frac{1}{2}}$$

$$i = 1, j = 1 \rightarrow (1+2r)U_{1,1}^{n+1} - rU_{1,0}^{n+1} - rU_{1,2}^{n+1} = (1-2r)U_{1,1}^{n+\frac{1}{2}} + rU_{0,1}^{n+\frac{1}{2}} + rU_{2,1}^{n+\frac{1}{2}}$$

$$i = 1, j = 2 \rightarrow (1+2r)U_{1,2}^{n+1} - rU_{1,1}^{n+1} - rU_{1,3}^{n+1} = (1-2r)U_{1,2}^{n+\frac{1}{2}} + rU_{0,2}^{n} + rU_{2,2}^{n+\frac{1}{2}}$$

....

....

$$i = 2, j = 0 \rightarrow (1+r)U_{2,0}^{n+1} - rU_{2,1}^{n+1} = (1-2r)U_{2,0}^{n+\frac{1}{2}} + rU_{1,0}^{n+\frac{1}{2}} + rU_{3,0}^{n+\frac{1}{2}}$$

$$i = 2, j = 1 \rightarrow (1+2r)U_{2,1}^{n+1} - rU_{2,0}^{n+1} - rU_{2,2}^{n+1} = (1-2r)U_{2,1}^{n+\frac{1}{2}} + rU_{1,1}^{n+\frac{1}{2}} + rU_{3,1}^{n+\frac{1}{2}}$$

$$i = 2, j = 2 \rightarrow (1+2r)U_{2,2}^{n+1} - rU_{2,1}^{n+1} - rU_{2,3}^{n+1} = (1-2r)U_{2,2}^{n+\frac{1}{2}} + rU_{1,2}^{n+\frac{1}{2}} + rU_{3,2}^{n+\frac{1}{2}}$$

....

....

Summing the left hand side will give $\sum_{i,j} U_{i,j}^{n+1}$ such that

$$U_{0,0}^{n+1} + U_{0,1}^{n+1} + U_{0,2}^{n+1} + \ldots\ldots\ldots\ldots U_{1,0}^{n+1} + U_{1,1}^{n+1} + U_{1,2}^{n+1} + \ldots\ldots\ldots\ldots U_{2,0}^{n+1} + U_{2,1}^{n+1} + U_{2,2}^{n+1} + \ldots\ldots\ldots\ldots$$
$$= \sum_{i,j} U_{i,j}^{n+1}$$

while the right hand side gives $\sum_{i,j} U_{i,j}^{n+\frac{1}{2}}$. Since

$$\sum_{i,j} U_{i,j}^{n} = \sum_{i,j} U_{i,j}^{n+\frac{1}{2}} = \sum_{i,j} U_{i,j}^{n+1}$$

We conclude that this equality holds for $\sum_{i,j} U_{i,j}^{0} = \sum_{i,j} U_{i,j}^{n}$ as well.

We used the code to verify this above by printing the summation of the solution at all gird points for different intermediate time steps (with fixed nx and ny). The results shown in Table **??** agrees with our proof.

Time	$\sum_{i,j} U_{i,j}$
0.0015625	1111.666239
0.0031250	1111.666239
0.0046875	1111.666239
0.00625	1111.666239
0.0078125	1111.666239
0.009375	1111.666239
0.0109375	1111.666239
0.0140625	1111.666239
0.015625	1111.666239
0.0171875	1111.666239
.....	....
.....	....
.....	....
0.996875	1111.666239
0.9984375	1111.666239
1	1111.666239

Table 1: Summing the solution over all grid points for different time steps (fixed nx and ny)

2 Problem No.2

2.1 Problem Description:

The FirzHugh-Nagumo equations

$$\frac{\partial v}{\partial t} = D\Delta v + (a - v)(v - 1)v - w + I$$
$$\frac{\partial w}{\partial t} = \epsilon(v - \gamma w)$$

are used in electrophysiology to model the cross membrane electrical potential (voltage) in cardiac tissue and in neurons. Assuming that the spatial coupling is local and passive results the term which looks like the diffusion of voltage. The state variable are the voltage v and recovery variable w.

1. Write a program to solve the FirzHugh-Nagumo equation on the unit square with homogeneous Neumann boundary conditions for v (meaning electrically insulated). Use a fractional step method to handle the diffusion and reactions separately. Use an ADI method for diffusion solve. Describe what ODE solver you used for the reactions and what fractional stepping you chose.

2. Use the following parameters $a = 0.1, \gamma = 2, \epsilon = 0.005, I = 0, D = 5 \times 10^{-5}$, and initial conditions

$$v(x, y, 0) = exp(-100(x^2 + y^2))$$
$$w(x, y, 0) = 0.0$$

Note that $v = 0, w = 0$ is a stable steady state of the system. Call this the rest state. For these initial conditions the voltage has been raised above in the bottom corner of the domain. Generate a numerical solution up to time $t = 300$. Visualize the voltage and describe the solution. Pick space and time steps to resolve the spatiotemporal dynamics of the solution you see. Discuss what the grid size and time step you used and why.

3. Use the same parameters from part (2), but use the initial conditions

$$v(x, y, 0) = 1 - 2x$$
$$w(x, y, 0) = 0.05y$$

and run the simulation until time t=600. Show the voltage at several points in time (pseudocolor plot, contour plot, or surface plot $z = V(x, y, t)$) and describe the solution.

2.2 Solution:

Part 1 & 2: FirzHugh-Nagumo equations can be solved using Strang splitting (second order accurate) such that the diffusion part is solved using ADI with time step Δt and the reaction part along with the second equation is solved using time steps of Δt. We start by solving the the reaction and the second equation which produce a non-linear coupled system of ODE which can be solved using RK4. We can write the equations as follows

$$\frac{dv}{dt} = f_v(v, w) = (a - v)(v - 1)v - w + I$$
$$\frac{dw}{dt} = f_w(v, w) = \epsilon(v - \gamma w)$$

The RK4 solver of this system is

$$v^{n+1} = v^n + \frac{1}{6}(kv_1 + 2kv_2 + 2kv_3 + kv_4), \qquad w^{n+1} = w^n + \frac{1}{6}(kw_1 + 2kw_2 + 2kw_3 + kw_4)$$

$$kv_1 = \frac{\Delta t}{2} f_v(v, w), \qquad kw_1 = \frac{\Delta t}{2} f_w(v, w)$$

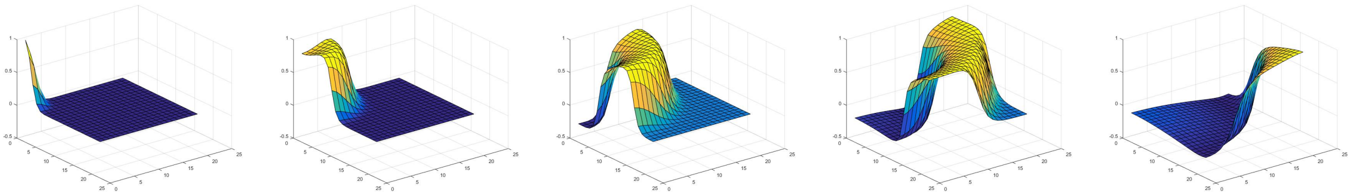
$$kv_2 = \frac{\Delta t}{2} f_v(v + \tfrac{1}{2}kv_1, w + \tfrac{1}{2}kw_1), \qquad kw_2 = \frac{\Delta t}{2} f_w(v + \tfrac{1}{2}kv_1, w + \tfrac{1}{2}kw_1)$$

$$kv_3 = \frac{\Delta t}{2} f_v(v + \tfrac{1}{2}kv_2, w + \tfrac{1}{2}kw_2), \qquad kw_3 = \frac{\Delta t}{2} f_w(v + \tfrac{1}{2}kv_2, w + \tfrac{1}{2}kw_2)$$

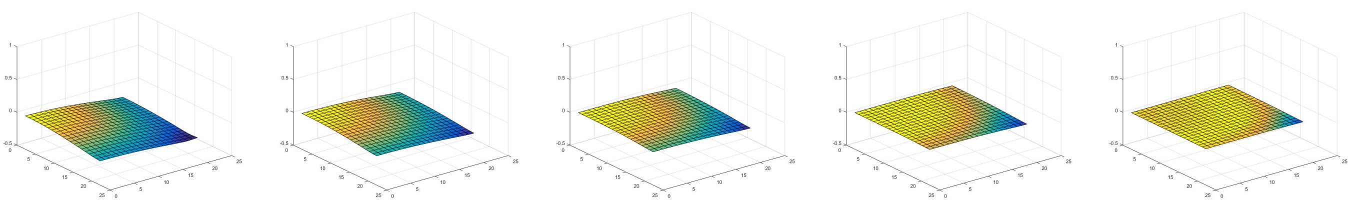
$$kv_4 = \frac{\Delta t}{2} f_v(v + kv_3, w + kw_3), \qquad kw_4 = \frac{\Delta t}{2} f_w(v + kv_3, w + kw_3)$$

After solving for half time step, we solve the diffusion part of the first equation using the same solver developed for Problem No.1 such that the value at step n is the values produced from the RK4 system. Following this, we solve the same RK4 system with same time step size ($\Delta t/2$) and the voltage values at time step n is the values produced by the ADI solver. The results for gird of size 20×20 with $\Delta t = 0.1$ is shown in Figure 3. The solution starts with a pump at the left bottom corner. The pump generates a wave that moves from the left bottom up all the way to right top corner. Following, the voltage turns back to the rest value (zero).

Several grid sizes has been tested (not shown) and all gives the same solution. We tested the following grid size expressed as $\Delta x \times \Delta t$: (0.05×1.0), (0.05×0.5), (0.05×0.1), (0.05×0.05) and (0.05×0.001). The one thing that changes using different time and space steps is the time it takes to get back to the rest state. But, at any gird and time step sizes, the solution is almost back to the rest state after 200 seconds.



(a) time=0 sec (b) time=33.3 sec (c) time=66.5 sec (d) time=99.9 sec (e) time=133.2 sec



(f) time=166.5 sec (g) time=199.8 sec (h) time=233.1 sec (i) time=266.4 sec (j) time=299.7 sec

Figure 3: Solution of FirzHugh-Nagumo equations with time and space of $\Delta t = 0.1, \Delta x = \Delta y = 0.05$ using Strang splitting and ADI for diffusion and RK4 for reaction and w and initial conditions as described in Part 2 and Neumann boundary conditions.

Part 3: Here we solve the same equation using the same system of equations and solvers but with different initial conditions and up to 600 seconds. The solution is shown in Figure 4.

With this initial conditions, the voltage represents a linear ramp function from [-1,1]. As the solution advances in time, the initial conditions turns into a spiral-like shape which seems to continue till the end of the simulation time (600 seconds).

We can see that both the first and second simulations share the phenomena of turning the initial conditions into wave-like shape that spreads across the domain. However, they differ in that the first was apple to vanish the effects of the initial conditions and turn to back to rest state.

Appendix

Here we show how to expand equations (2) and (3) from Problem (1) and the matrix formulation. Starting with z-sweep with a grid of size $nx \times ny$, where i and j represent numbering for the numerical grid points, $r = 0.1\Delta t/2h^2$, $\alpha = 1 + r$ and $\beta = 1 + 2r$, we obtain:

For $j = 0$

$$i = 0, j = 0 \rightarrow (1+r)U_{0,0}^{n+\frac{1}{2}} - rU_{1,0}^{n+\frac{1}{2}} = (1-r)U_{0,0}^{n} + rU_{0,1}^{n}$$

$$i = 1, j = 0 \rightarrow (1+2r)U_{1,0}^{n+\frac{1}{2}} - rU_{0,0}^{n+\frac{1}{2}} - rU_{2,0}^{n+\frac{1}{2}} = (1-r)U_{1,0}^{n} + rU_{1,1}^{n}$$

$$i = 2, j = 0 \rightarrow (1+2r)U_{2,0}^{n+\frac{1}{2}} - rU_{1,0}^{n+\frac{1}{2}} - rU_{3,0}^{n+\frac{1}{2}} = (1-r)U_{2,0}^{n} + rU_{2,1}^{n}$$

....
....
....

$$i = nx-1, j = 0 \rightarrow (1+r)U_{nx-1,0}^{n+\frac{1}{2}} - rU_{nx-2,0}^{n+\frac{1}{2}} = (1-r)U_{nx-1,0}^{n} + rU_{nx-1,1}^{n}$$

$$\begin{vmatrix} \alpha & -r & 0 & ... & ... & ... & 0 & 0 \\ -r & \beta & -r & 0 & ... & ... & 0 & 0 \\ 0 & -r & \beta & -r & 0 & ... & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & -r & \beta & -r \\ ... & ... & ... & ... & ... & ... & -r & \alpha \end{vmatrix} \begin{vmatrix} U_{0,0}^{n+\frac{1}{2}} \\ U_{1,0}^{n+\frac{1}{2}} \\ U_{2,0}^{n+\frac{1}{2}} \\ ... \\ ... \\ ... \\ U_{nx-2,0}^{n+\frac{1}{2}} \\ U_{nx-1,0}^{n+\frac{1}{2}} \end{vmatrix} = \begin{vmatrix} (1-r)U_{0,0}^{n} + rU_{0,1}^{n} \\ (1-r)U_{1,0}^{n} + rU_{1,1}^{n} \\ (1-r)U_{2,0}^{n} + rU_{2,1}^{n} \\ ... \\ ... \\ ... \\ (1-r)U_{nx-2,0}^{n} + rU_{nx-2,1}^{n} \\ (1-r)U_{nx-1,0}^{n} + rU_{nx-1,1}^{n} \end{vmatrix}$$

For $j = 1$

$$i = 0, j = 1 \rightarrow (1+r)U_{0,1}^{n+\frac{1}{2}} - rU_{1,1}^{n+\frac{1}{2}} = (1-2r)U_{0,1}^{n} + rU_{0,0}^{n} + rU_{0,2}^{n}$$

$$i = 1, j = 1 \rightarrow (1+2r)U_{1,1}^{n+\frac{1}{2}} - rU_{0,1}^{n+\frac{1}{2}} - rU_{2,1}^{n+\frac{1}{2}} = (1-2r)U_{1,1}^{n} + rU_{1,0}^{n} + rU_{1,2}^{n}$$

$$i = 2, j = 1 \rightarrow (1+2r)U_{2,1}^{n+\frac{1}{2}} - rU_{1,1}^{n+\frac{1}{2}} - rU_{3,1}^{n+\frac{1}{2}} = (1-2r)U_{2,1}^{n} + rU_{2,0}^{n} + rU_{2,2}^{n}$$

....
....
....

$$i = nx-1, j = 1 \rightarrow (1+r)U_{nx-1,1}^{n+\frac{1}{2}} - rU_{nx-2,1}^{n+\frac{1}{2}} = (1-2r)U_{nx-1,1}^{n} + rU_{nx-1,0}^{n} + rU_{nx-1,2}^{n}$$

$$\begin{vmatrix} \alpha & -r & 0 & ... & ... & ... & 0 & 0 \\ -r & \beta & -r & 0 & ... & ... & 0 & 0 \\ 0 & -r & \beta & -r & 0 & ... & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & -r & \beta & -r \\ ... & ... & ... & ... & ... & ... & -r & \alpha \end{vmatrix} \begin{vmatrix} U_{0,1}^{n+\frac{1}{2}} \\ U_{1,1}^{n+\frac{1}{2}} \\ U_{2,1}^{n+\frac{1}{2}} \\ ... \\ ... \\ ... \\ U_{nx-2,1}^{n+\frac{1}{2}} \\ U_{nx-1,1}^{n+\frac{1}{2}} \end{vmatrix} = \begin{vmatrix} (1-2r)U_{0,1}^{n} + rU_{0,0}^{n} + rU_{0,2}^{n} \\ (1-2r)U_{1,1}^{n} + rU_{1,0}^{n} + rU_{1,2}^{n} \\ (1-2r)U_{2,1}^{n} + rU_{2,0}^{n} + rU_{2,2}^{n} \\ ... \\ ... \\ ... \\ (1-2r)U_{nx-2,1}^{n} + rU_{nx-2,0}^{n} + rU_{nx-2,2}^{n} \\ (1-2r)U_{nx-1,1}^{n} + rU_{nx-1,0}^{n} + rU_{nx-1,2}^{n} \end{vmatrix}$$

For $j = 2$

$$i = 0, j = 2 \rightarrow (1+r)U_{0,2}^{n+\frac{1}{2}} - rU_{1,2}^{n+\frac{1}{2}} = (1-2r)U_{0,2}^{n} + rU_{0,1}^{n} + rU_{0,3}^{n}$$

$$i = 1, j = 2 \rightarrow (1+2r)U_{1,2}^{n+\frac{1}{2}} - rU_{0,2}^{n+\frac{1}{2}} - rU_{2,2}^{n+\frac{1}{2}} = (1-2r)U_{1,2}^{n} + rU_{1,1}^{n} + rU_{1,3}^{n}$$

$$i = 2, j = 2 \rightarrow (1+2r)U_{2,2}^{n+\frac{1}{2}} - rU_{1,2}^{n+\frac{1}{2}} - rU_{3,2}^{n+\frac{1}{2}} = (1-2r)U_{2,2}^{n} + rU_{2,1}^{n} + rU_{2,3}^{n}$$

....
....
....

$$i = nx-1, j = 2 \rightarrow (1+r)U_{nx-1,2}^{n+\frac{1}{2}} - rU_{nx-2,2}^{n+\frac{1}{2}} = (1-2r)U_{nx-1,2}^{n} + rU_{nx-1,1}^{n} + rU_{nx-1,3}^{n}$$

$$\begin{vmatrix} \alpha & -r & 0 & ... & ... & ... & 0 & 0 \\ -r & \beta & -r & 0 & ... & ... & 0 & 0 \\ 0 & -r & \beta & -r & 0 & ... & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & -r & \beta & -r \\ ... & ... & ... & ... & ... & ... & -r & \alpha \end{vmatrix} \begin{vmatrix} U_{0,2}^{n+\frac{1}{2}} \\ U_{1,2}^{n+\frac{1}{2}} \\ U_{2,2}^{n+\frac{1}{2}} \\ ... \\ ... \\ ... \\ U_{nx-2,2}^{n+\frac{1}{2}} \\ U_{nx-1,2}^{n+\frac{1}{2}} \end{vmatrix} = \begin{vmatrix} (1-2r)U_{0,2}^{n} + rU_{0,1}^{n} + rU_{0,3}^{n} \\ (1-2r)U_{1,2}^{n} + rU_{1,1}^{n} + rU_{1,3}^{n} \\ (1-2r)U_{2,2}^{n} + rU_{2,1}^{n} + rU_{2,3}^{n} \\ ... \\ ... \\ ... \\ (1-2r)U_{nx-2,2}^{n} + rU_{nx-2,1}^{n} + rU_{nx-2,3}^{n} \\ (1-2r)U_{nx-1,2}^{n} + rU_{nx-1,1}^{n} + rU_{nx-1,3}^{n} \end{vmatrix}$$

For $j = ny - 1$

$$i = 0, j = ny-1 \rightarrow (1+r)U^{n+\frac{1}{2}}_{0,ny-1} - rU^{n+\frac{1}{2}}_{1,ny-1} = (1-r)U^{n}_{0,ny-1} + rU^{n}_{0,ny-2}$$

$$i = 1, j = ny-1 \rightarrow (1+2r)U^{n+\frac{1}{2}}_{1,ny-1} - rU^{n+\frac{1}{2}}_{0,ny-1} - rU^{n+\frac{1}{2}}_{2,ny-1} = (1-r)U^{n}_{1,2} + rU^{n}_{1,ny-2}$$

$$i = 2, j = ny-1 \rightarrow (1+2r)U^{n+\frac{1}{2}}_{2,ny-1} - rU^{n+\frac{1}{2}}_{1,ny-1} - rU^{n+\frac{1}{2}}_{3,ny-1} = (1-r)U^{n}_{2,2} + rU^{n}_{2,ny-2}$$

....
....
....

$$i = nx-1, j = ny-1 \rightarrow (1+r)U^{n+\frac{1}{2}}_{nx-1,ny-1} - rU^{n+\frac{1}{2}}_{nx-2,ny-1} = (1-r)U^{n}_{nx-1,ny-1} + rU^{n}_{nx-1,ny-2}$$

$$\begin{vmatrix} \alpha & -r & 0 & ... & ... & ... & 0 & 0 \\ -r & \beta & -r & 0 & ... & ... & 0 & 0 \\ 0 & -r & \beta & -r & 0 & ... & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & -r & \beta & -r \\ ... & ... & ... & ... & ... & ... & -r & \alpha \end{vmatrix} \begin{vmatrix} U^{n+\frac{1}{2}}_{0,ny-1} \\ U^{n+\frac{1}{2}}_{1,ny-1} \\ U^{n+\frac{1}{2}}_{2,ny-1} \\ ... \\ ... \\ ... \\ U^{n+\frac{1}{2}}_{nx-2,ny-1} \\ U^{n+\frac{1}{2}}_{nx-1,ny-1} \end{vmatrix} = \begin{vmatrix} (1-r)U^{n}_{0,ny-1} + rU^{n}_{0,ny-2} \\ (1-r)U^{n}_{1,ny-1} + rU^{n}_{1,ny-2} \\ (1-r)U^{n}_{2,ny-1} + rU^{n}_{2,ny-2} \\ ... \\ ... \\ ... \\ (1-r)U^{n}_{nx-2,ny-1} + rU^{n}_{nx-2,ny-2} \\ (1-r)U^{n}_{nx-1,ny-1} + rU^{n}_{nx-1,ny-2} \end{vmatrix}$$

We can carry on the same operations for the y-sweep but each matrix is formulated for different i. For example, for $i = 0$

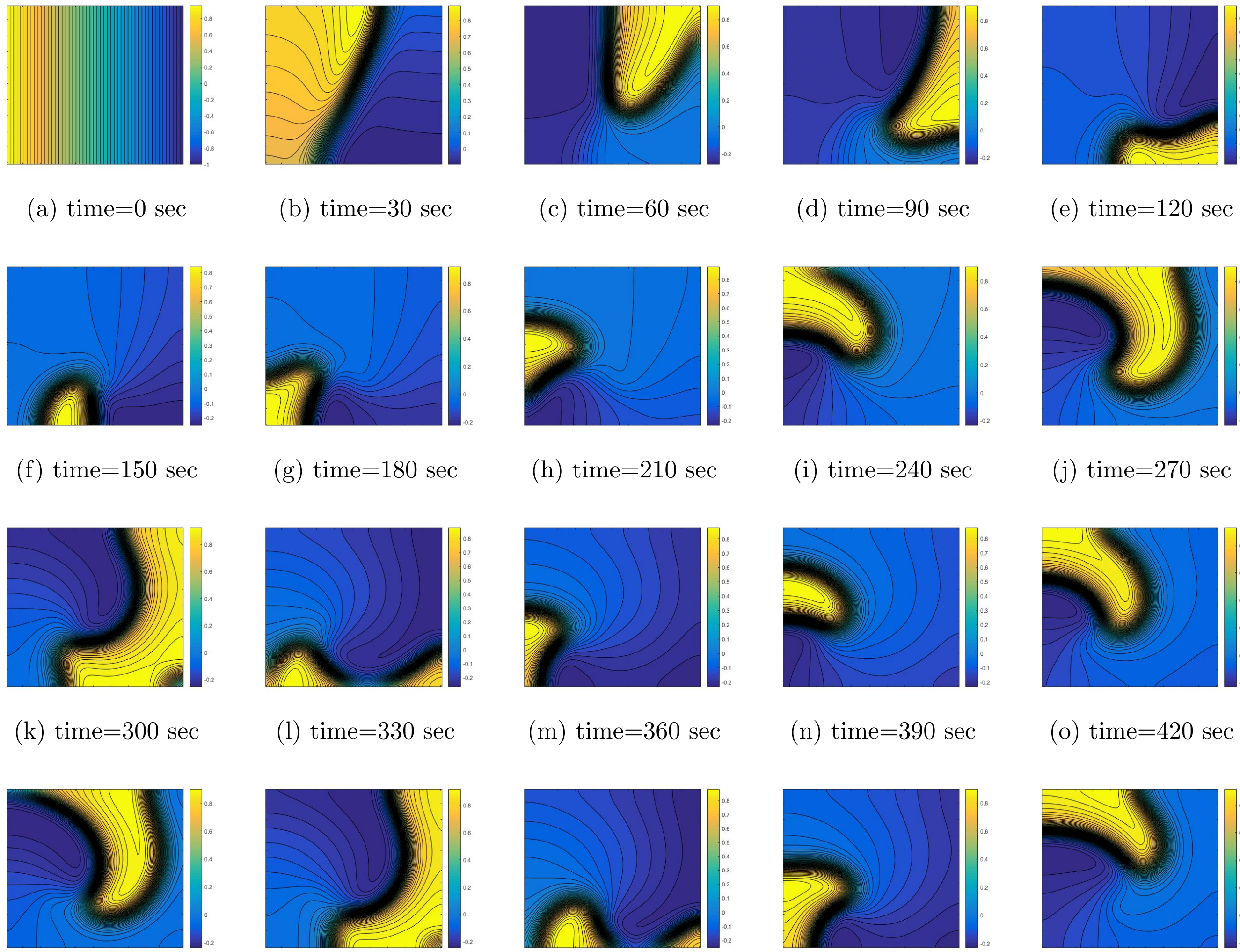
$$\begin{vmatrix} \alpha & -r & 0 & ... & ... & ... & 0 & 0 \\ -r & \beta & -r & 0 & ... & ... & 0 & 0 \\ 0 & -r & \beta & -r & 0 & ... & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & -r & \beta & -r \\ ... & ... & ... & ... & ... & ... & -r & \alpha \end{vmatrix} \begin{vmatrix} U^{n+1}_{0,0} \\ U^{n+1}_{0,1} \\ U^{n+1}_{0,2} \\ ... \\ ... \\ ... \\ U^{n+1}_{0,ny-2} \\ U^{n+1}_{0,ny-1} \end{vmatrix} = \begin{vmatrix} (1-r)U^{n+\frac{1}{2}}_{0,0} + rU^{n+\frac{1}{2}}_{1,0} \\ (1-r)U^{n+\frac{1}{2}}_{0,1} + rU^{n+\frac{1}{2}}_{1,1} \\ (1-r)U^{n+\frac{1}{2}}_{0,2} + rU^{n+\frac{1}{2}}_{1,2} \\ ... \\ ... \\ ... \\ (1-r)U^{n+\frac{1}{2}}_{0,ny-2} + rU^{n+\frac{1}{2}}_{1,ny-2} \\ (1-r)U^{n+\frac{1}{2}}_{0,ny-1} + rU^{n+\frac{1}{2}}_{1,ny-1} \end{vmatrix}$$

For $i = 1$

$$\begin{vmatrix} \alpha & -r & 0 & ... & ... & ... & 0 & 0 \\ -r & \beta & -r & 0 & ... & ... & 0 & 0 \\ 0 & -r & \beta & -r & 0 & ... & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & -r & \beta & -r \\ ... & ... & ... & ... & ... & ... & -r & \alpha \end{vmatrix} \begin{vmatrix} U_{1,0}^{n+1} \\ U_{1,1}^{n+1} \\ U_{1,2}^{n+1} \\ ... \\ ... \\ ... \\ U_{1,ny-2}^{n+1} \\ U_{1,ny-1}^{n+1} \end{vmatrix} = \begin{vmatrix} (1-2r)U_{1,0}^{n+\frac{1}{2}} + rU_{0,0}^{n+\frac{1}{2}} + rU_{2,0}^{n+\frac{1}{2}} \\ (1-2r)U_{1,1}^{n+\frac{1}{2}} + rU_{0,1}^{n+\frac{1}{2}} + rU_{2,1}^{n+\frac{1}{2}} \\ (1-2r)U_{1,2}^{n+\frac{1}{2}} + rU_{0,2}^{n+\frac{1}{2}} + rU_{2,2}^{n+\frac{1}{2}} \\ ... \\ ... \\ ... \\ (1-2r)U_{1,ny-2}^{n+\frac{1}{2}} + rU_{0,ny-2}^{n+\frac{1}{2}} + rU_{2,ny-2}^{n+\frac{1}{2}} \\ (1-2r)U_{1,ny-1}^{n+\frac{1}{2}} + rU_{0,ny-1}^{n+\frac{1}{2}} + rU_{2,ny-1}^{n+\frac{1}{2}} \end{vmatrix}$$

For $nx - 1$

$$\begin{vmatrix} \alpha & -r & 0 & ... & ... & ... & 0 & 0 \\ -r & \beta & -r & 0 & ... & ... & 0 & 0 \\ 0 & -r & \beta & -r & 0 & ... & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & -r & \beta & -r \\ ... & ... & ... & ... & ... & ... & -r & \alpha \end{vmatrix} \begin{vmatrix} U_{nx-1,0}^{n+1} \\ U_{nx-1,1}^{n+1} \\ U_{nx-1,2}^{n+1} \\ ... \\ ... \\ ... \\ U_{nx-1,ny-2}^{n+1} \\ U_{nx-1,ny-1}^{n+1} \end{vmatrix} = \begin{vmatrix} (1-r)U_{nx-1,0}^{n+\frac{1}{2}} + rU_{nx-2,0}^{n+\frac{1}{2}} \\ (1-r)U_{nx-1,1}^{n+\frac{1}{2}} + rU_{nx-2,1}^{n+\frac{1}{2}} \\ (1-r)U_{nx-1,2}^{n+\frac{1}{2}} + rU_{nx-2,2}^{n+\frac{1}{2}} \\ ... \\ ... \\ ... \\ (1-r)U_{nx-1,ny-2}^{n+\frac{1}{2}} + rU_{nx-2,ny-2}^{n+\frac{1}{2}} \\ (1-r)U_{nx-1,ny-1}^{n+\frac{1}{2}} + rU_{nx-2,ny-1}^{n+\frac{1}{2}} \end{vmatrix}$$



(a) time=0 sec (b) time=30 sec (c) time=60 sec (d) time=90 sec (e) time=120 sec

(f) time=150 sec (g) time=180 sec (h) time=210 sec (i) time=240 sec (j) time=270 sec

(k) time=300 sec (l) time=330 sec (m) time=360 sec (n) time=390 sec (o) time=420 sec

(p) time=450 sec (q) time=480 sec (r) time=510 sec (s) time=540 sec (t) time=600 sec

Figure 4: Solution of FirzHugh-Nagumo equations with time and space of $\Delta t = 0.1, \Delta x = \Delta y = 0.05$ using Strang splitting and ADI for diffusion and RK4 for reaction and w and initial conditions as described in Part 3 and Neumann boundary conditions.