Assignment 7:

Question2, Implementation:

The explicit scheme used in class for the heat equation $u_t = c^2 u_{tt}$ is

$$\frac{u_i^{j+1} - u_i^j}{\Delta t} = c^2 \frac{u_{i-1}^j - 2u_i^j + 2u_{i+1}^j}{\Delta x^2}$$

Since c = 1, solving for u_i^{j+1} the equation can be rewritten as below:

$$u_i^{j+1} = u_i^j + r\left(u_{i-1}^j - 2u_i^j + 2u_{i+1}^j\right) \text{ where } r = \frac{\Delta t}{\Delta x^2}$$

Creating a system of matrices, excluding $u_0^j = u(0,t) = 0$ and $u_N^j = u(1,t) = 0$ from the first and last rows, will be the below matrix: (Sorry for the screenshot ☹)

$$\overrightarrow{u}_i^{j+1} - \begin{bmatrix} ru_0^j \\ 0 \\ \vdots \\ 0 \\ ru_N^j \end{bmatrix} = \begin{bmatrix} 1-2r & r & 0 & & \\ r & 1-2r & r & & \\ 0 & r & 1-2r & \ddots & \\ & & \ddots & \ddots & r \\ & & & r & 1-2r \end{bmatrix} \begin{bmatrix} u_1^j \\ u_2^j \\ \vdots \\ u_{N-2}^j \\ u_{N-1}^j \end{bmatrix}$$

With these boundary conditions $u_0^j = u(0,t) = 0$ and $u_N^j = u(1,t) = 0$ for $t \in [0, 0.1]$, $r_0^j = u(0,t) = 0$ and $r_N^j = u(1,t) = 0$ will be deleted and resulting the below matrix:

$$\overrightarrow{u}_i^{j+1} = \begin{bmatrix} 1-2r & r & 0 & & \\ r & 1-2r & r & & \\ 0 & r & 1-2r & \ddots & \\ & & \ddots & \ddots & r \\ & & & r & 1-2r \end{bmatrix} \begin{bmatrix} u_1^j \\ u_2^j \\ \vdots \\ u_{N-2}^j \\ u_{N-1}^j \end{bmatrix}$$

I have chosen $dx\_values = [0.2, 0.05, 0.025, 0.0125]$, and to scheme to be conditionally stable, I chose $dt = 0.00007812$, to have r = 0.5 for the smallest dx to make sure that for bigger dx the condition is again correct.

$$\Delta t = \frac{0.0125^2}{4}, so \rightarrow \; r = \frac{\Delta t}{\Delta x^2} \leq 0.5$$

The slope (order of convergence) of this method which is calculated in the code is 1.9270, which is near 2 and as we expected it.

The numerical solution is less accurate than the exact solution, but it provides a good approximation of the exact solution. The error in the numerical solution can be reduced by decreasing Δx and Δt. However, this will increase the computational cost of the solution. As dx decreases, the spatial grid becomes finer, allowing for a more accurate representation of the solution. However, this comes at the cost of increased computational requirements. The slope of the log-log plot of errors vs dx represents the order of convergence. The slope is close to 2, which suggests that the method is second-order accurate. In other words, as dx is halved, the error decreases by a factor of 2. This is a common result for explicit finite difference methods applied to parabolic PDEs like the heat equation. Achieving a higher order of accuracy often requires more sophisticated numerical methods or finer discretization, which can increase computational costs. In practice, choosing an appropriate balance between accuracy and computational efficiency is a common consideration.

Now, the numerical vs. exact solution at t = 0.1 and two different dx: 0.2, 0.025 are plotted to compare them. As the dx decreases, the numerical solution will be closer to the exact solution which is as expected.

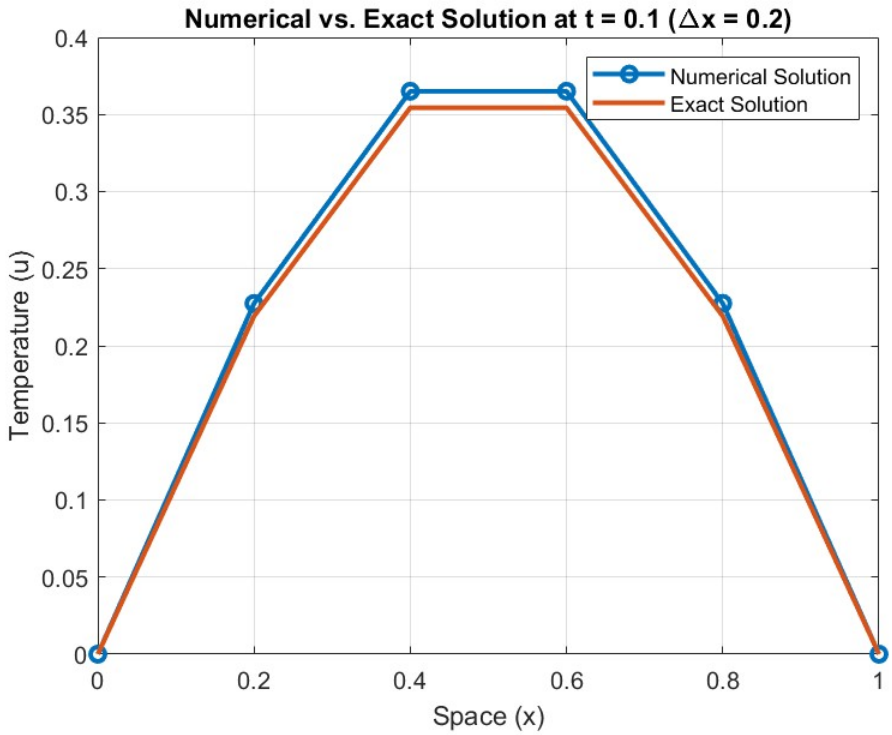


Figure. Numerical vs. Exact solution at t = 0.1, dx = 0.2

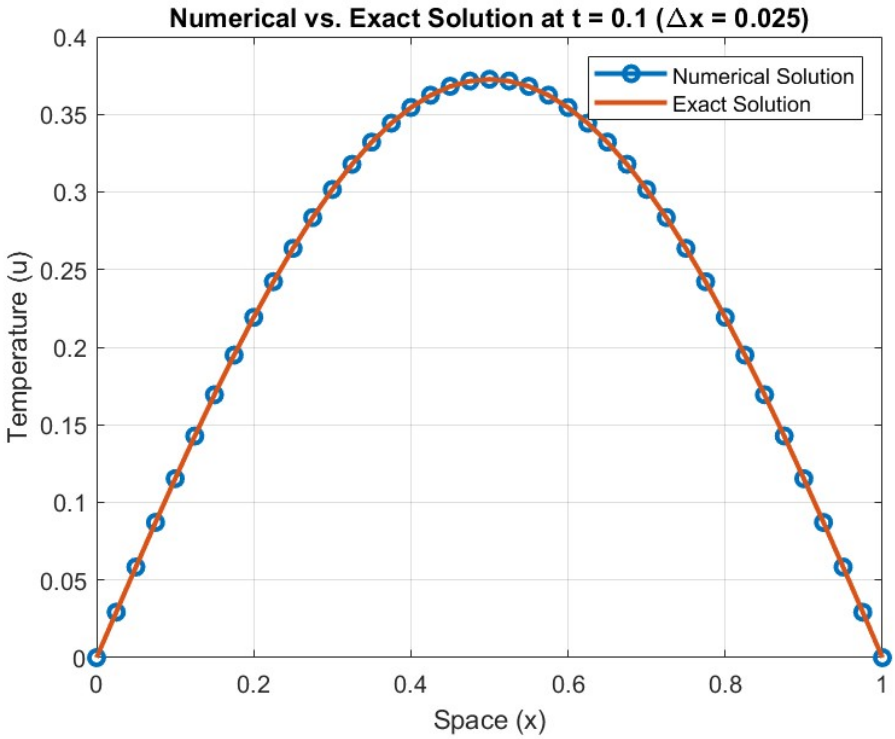
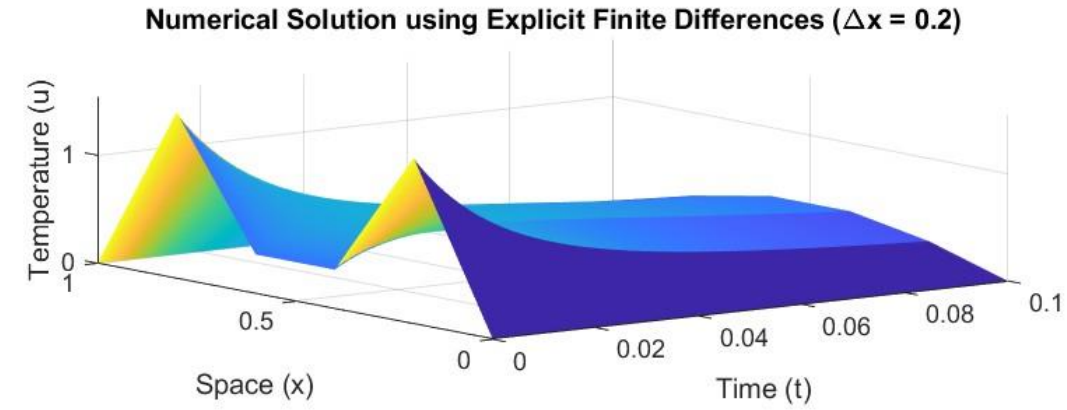
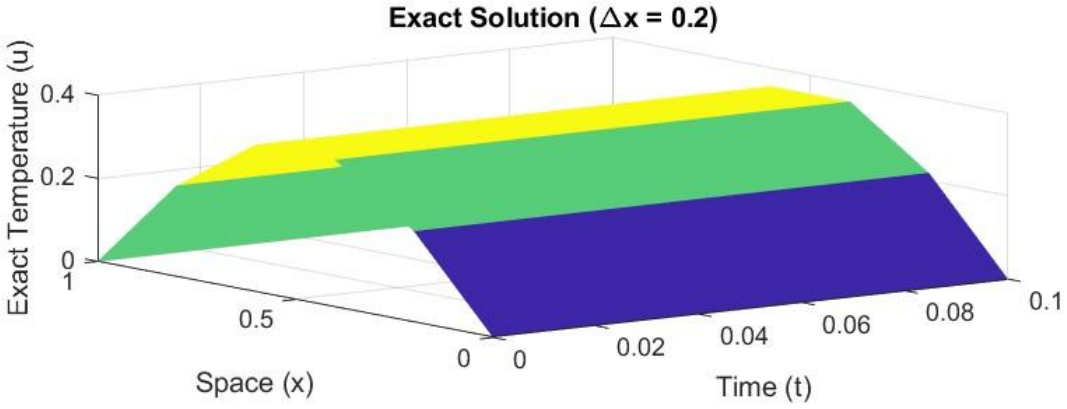
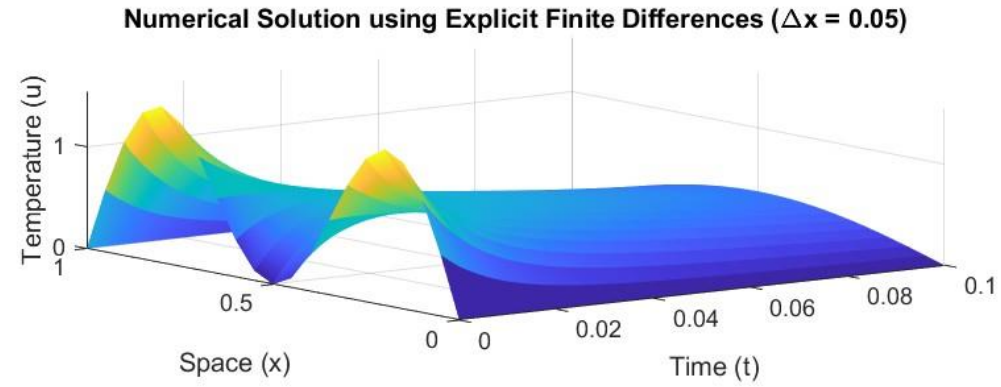


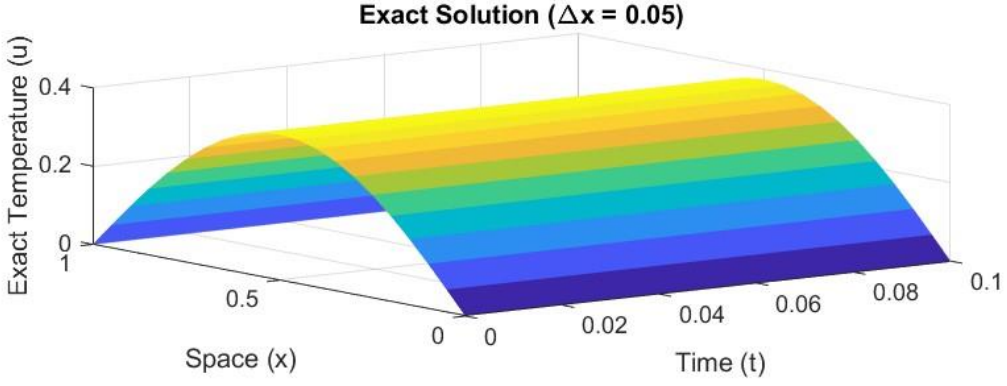
Figure. Numerical vs. Exact solution at t = 0.1, dx = 0.025

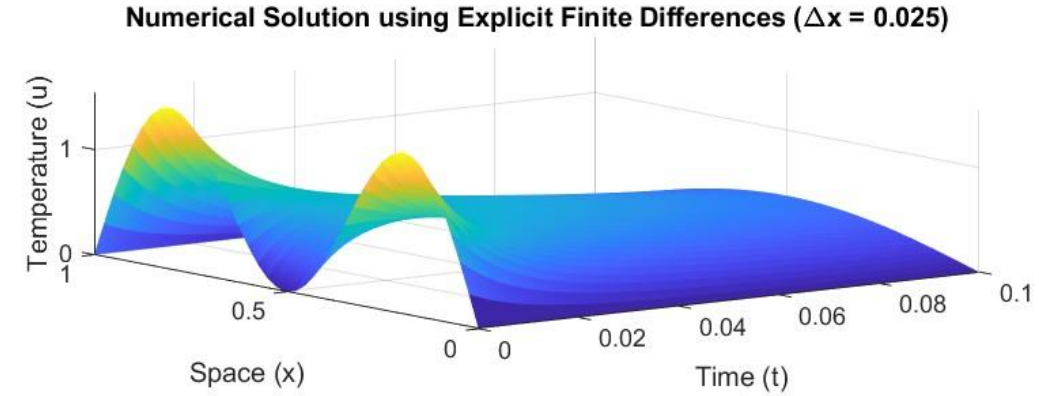
The next plots are surf graphs exact solutions and numerical solutions at different dx sizes, which as you mentioned may not be very helpful. Although, we can observe as the dx decreases the numerical solution will be more smooth and closer to the exact solution.

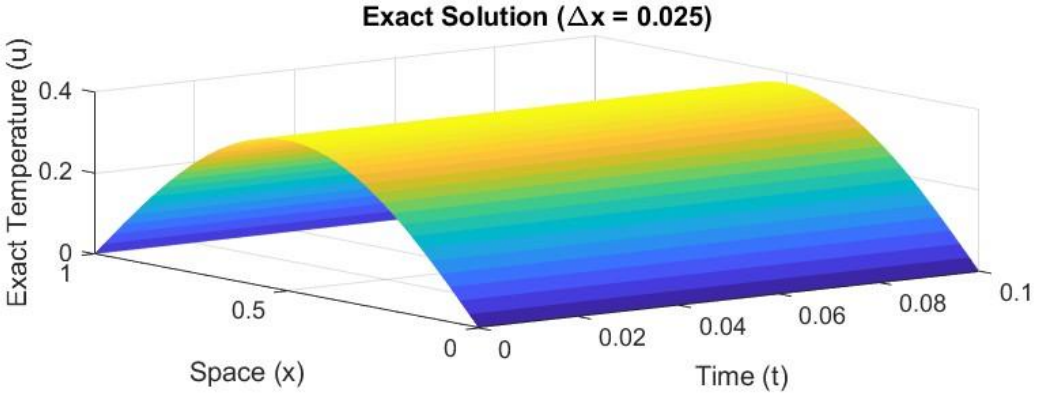

Numerical Solution using Explicit Finite Differences (Δx = 0.2)
Temperature (u)
1
0
1
0.5
0
Space (x)
0
0.02
0.04
0.06
0.08
0.1
Time (t)

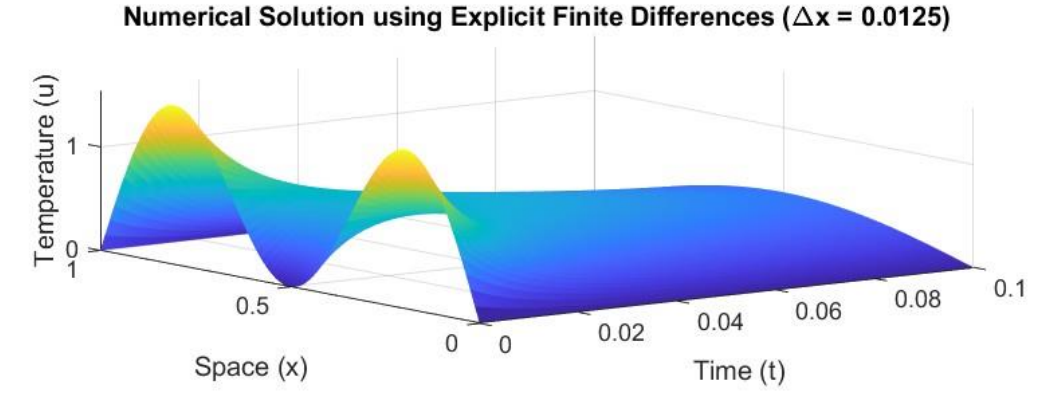

Exact Solution (Δx = 0.2)
Exact Temperature (u)
0.4
0.2
0
1
0.5
0
Space (x)
0
0.02
0.04
0.06
0.08
0.1
Time (t)

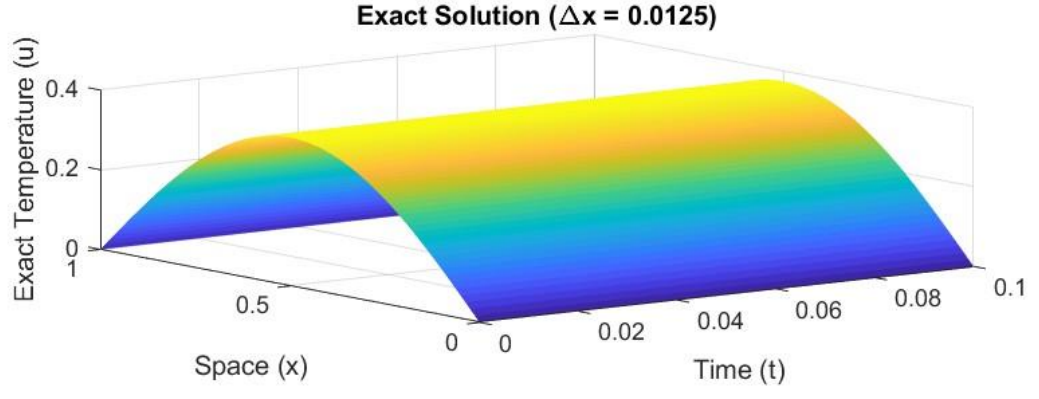

Numerical Solution using Explicit Finite Differences (Δx = 0.05)
Temperature (u)
1
0
1
0.5
0
Space (x)
0
0.02
0.04
0.06
0.08
0.1
Time (t)


Exact Solution (Δx = 0.05)
Exact Temperature (u)
0.4
0.2
0
1
0.5
0
Space (x)
0
0.02
0.04
0.06
0.08
0.1
Time (t)


Numerical Solution using Explicit Finite Differences (Δx = 0.025)
Temperature (u)
1
0
1
0.5
0
Space (x)
0
0.02
0.04
0.06
0.08
0.1
Time (t)


Exact Solution (Δx = 0.025)
Exact Temperature (u)
0.4
0.2
0
1
0.5
0
Space (x)
0
0.02
0.04
0.06
0.08
0.1
Time (t)


Numerical Solution using Explicit Finite Differences (Δx = 0.0125)
Temperature (u)
1
0
1
0.5
0
Space (x)
0
0.02
0.04
0.06
0.08
0.1
Time (t)


Exact Solution (Δx = 0.0125)
Exact Temperature (u)
0.4
0.2
0
1
0.5
0
Space (x)
0
0.02
0.04
0.06
0.08
0.1
Time (t)

Error Analysis:

For this problem, I chose $dx\_values = [0.2, 0.05, 0.025, 0.0125]$ to have a more refined error calculation. I also chose $dt = 0.00007812$, based on the Von Neumann stability criterion:

$$\Delta t = \frac{0.0125^2}{4}, so \rightarrow \ r = \frac{\Delta t}{\Delta x^2} \leq 0.5$$

A log-log plot showing the error of the scheme with different Δx is shown below. I am not sure if the errors are correct or not especially for the error at dx = 0.0125, because I think it should be less than the error at dx = 0.025. The good thing is that the order of spatial accuracy is 2, and by decreasing the dx, the error decreases.

Error @ dx=0.2	Error @ dx=0.05	Error @ dx=0.025	Error @ dx=0.0125
0.0106631428323490	0.000595139327888516	4.58749344580123e-05	9.17284263366458e-05

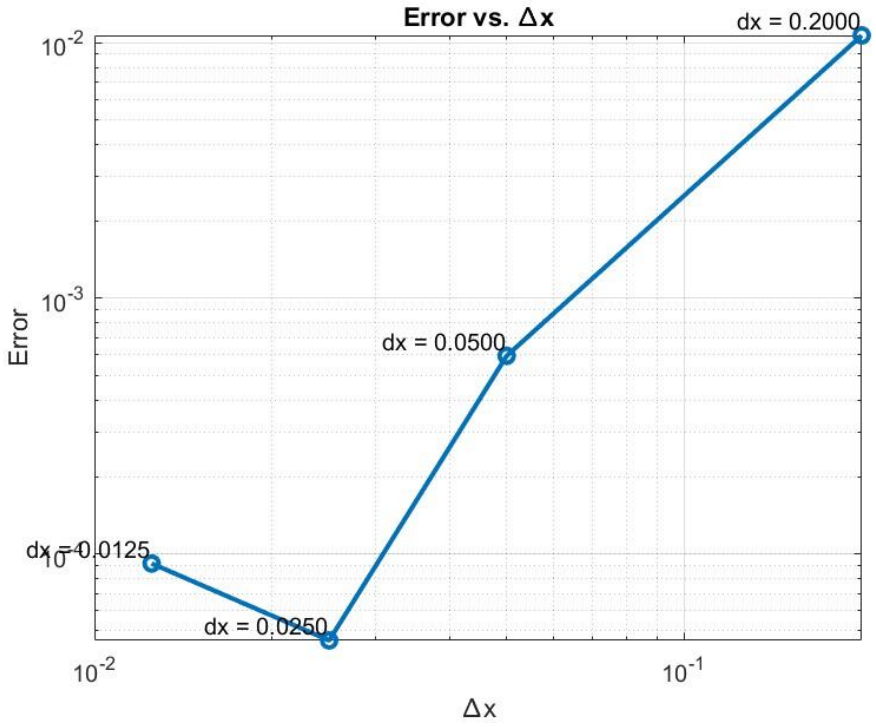


Figure: Question 2 LogLog Plot of Error

The "kink" or initial spike in error is a common behavior when using explicit finite difference schemes for solving partial differential equations (PDEs). This behavior is often associated with the stability and accuracy of the numerical method, especially for explicit methods where the time step and spatial discretization play crucial roles.

The possible explanations for this may be because explicit finite difference schemes have stability conditions, and violating these conditions can lead to numerical instability. The initial spike might

be related to the initial conditions and the chosen time step size. Another reason is that maybe the choice of spatial discretization and time step affects the numerical dispersion of the method.

We can try reducing the time step or refining the spatial grid to see if the initial spike diminishes. However, this might increase the computational cost.