A REPORT ON:

NUMERICAL METHODS FOR PARTIAL DIFFERENTIAL EQUATIONS

BY

Shaheen Ali

ID – 2021B4A33044H

In fulfillment of

MATH F266 - Study Project

Instructor: Dr. K Bhargav Kumar

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI, HYDERABAD CAMPUS

FIRST SEMESTER 2024-2025

ACKNOWLEDGEMENTS

We sincerely thank BITS Pilani Hyderabad campus for providing us with this wonderful opportunity to undertake an independent study project. The university's continued commitment to facilitating student experiential learning is deeply appreciated. We are extremely grateful to our project mentor, Dr. K Bhargav Kumar, for his invaluable guidance and support throughout this project. Bhargav Sir went above and beyond to ensure our success, holding weekly review meetings to provide practical feedback and help refine our approach. His suggestions were instrumental in shaping our work. Thanks to his motivation, problem-solving assistance, and the university enabling remote project work, we could complete this cherished learning experience. This has been an enriching learning journey under Bhargav Sir's mentorship. We feel privileged to have had this opportunity, which was enabled by BITS Pilani Hyderabad.

Birla Institute of Technology and Science-Pilani,

Hyderabad Campus

Certificate

This is to certify that the project report titled **"NUMERICAL METHODS FOR PARTIAL DIFFERENTIAL EQUATIONS"** submitted by

Shaheen Ali- 2021B4A33044H

In partial fulfilment of the requirements of the course MATH F266, the Study Project Course comprises the work they have done under my supervision and guidance.

Date: 28th November 2024

(Dr. K BHARGAV KUMAR)

BITS- Pilani, Hyderabad Campus

ABSTRACT

Partial differential equations (PDEs) are fundamental in modelling dynamic physical processes, such as heat transfer and particle transport, across a wide range of scientific and engineering fields. However, the complexity of real-world problems often limits the availability of analytical solutions, making numerical methods crucial for obtaining approximate solutions. In this project, we employ finite difference methods (FDM) to solve the heat and transport equations by discretizing space and time into a computational grid.

We implement and evaluate different finite difference schemes, including explicit (FTFS and FTBS for the transport equation, FTCS for the heat equation as well as the Lax Wendroff equation) and implicit (BTCS for the heat equation, BTFS and BTBS for the heat equation) methods, comparing their accuracy, stability, and computational efficiency performance. We assess how these factors influence solution accuracy by varying grid resolutions and time steps. Additionally, we compare the numerical solution of the heat equation with its analytical counterpart to evaluate the effectiveness of the numerical approach. This study provides insights into the trade-offs between computational cost and accuracy, emphasizing the importance of stability, consistency, and convergence in selecting appropriate numerical techniques for solving complex PDEs in practical scenarios.

Table of Contents

OBJECTIVES

1. **Runge-Kutta Methods & Finite Difference Schemes**: Study Runge-Kutta methods for solving ODEs and explore finite difference schemes to transform PDEs into discrete forms for numerical solutions.

2. **Transport & Heat Equation Schemes**: Implement and analyze FTFS and FTBS schemes for the transport equation, compare stability, and solve the heat equation using the FTCS scheme with varying parameters, optimizing for accuracy and efficiency.

3. **Error Analysis & Stability**: Compare numerical solutions to analytical ones, and assess the consistency, convergence, and stability of schemes using the Lax Equivalence Theorem.

4. **Explicit Schemes**: Implement the Lax-Wendroff scheme for the transport equation and the heat equation and compare stability with varying parameters.

5. **Implicit Schemes:** Implement the Crank Nicolson scheme for the transport equation, compare stability and optimize for accuracy and efficiency.

INTRODUCTION

Partial differential equations (PDEs) are fundamental mathematical tools for describing a wide range of physical phenomena. Some examples include: the distribution of heat within materials, the flow of fluids and particles, and how they disperse through space in time. Among the most common PDES is the heat equation which describes how temperature evolves within some region in space as a function of time and the transport equation which describes how some substance gets advected due to a fluid or, more generally, through porous media. Consequently, PDEs are foundational concepts in areas like physics, engineering and environmental science.

For certain simple geometry configurations and boundary conditions it is possible to obtain an exact solution to these PDEs analytically but real life problems often involve complex geometries non-linear phenomena that render impossible or very hard an analytical treatment for studying such systems. Numerically solving this sets of equations provides an alternative path for their study when no analytical answer can be found or even in order to only approximate it with prescribed accuracy. The Finite Difference Method (FDM) is one among several if not probably the most popular algorithm used by scientists and engineers to address PDES numerically essentially by discretizing space into grids over which numerical operations are carried out at each time step .

In the finite difference approach, the solution domain is divided into a grid of points, and a set of discrete points on this grid replaces the continuous spatial and temporal variables in the differential equations. The grid forms a lattice where the approximate solution is computed iteratively. The derivatives occurring in the PDE are approximated by difference equations with unknowns that depend on both neighbouring time and space nodes. For instance, to approximate first order derivatives (with respect to either time or space), we may use for each node pair surrounding an origin another pair further away from it; thus Lele (1992) provided two comprehensive reviews on higher-order finite difference schemes many years ago.

The accuracy and efficiency of a finite difference scheme depends on the grid resolution chosen and the type of finite difference scheme. This project will look into explicit and implicit finite differences schemes for solving the heat equation as well as the transport equation. In explicit method, at each time step the solution is directly calculated from values known from the previous time-step. Since it’s straight-forward to implement, one problem with Explicit methods is that they usually require small time-steps to be stable (satisfying Courant Condition). Implicit methods overcome this problem by making use of iterative schemes to solve a set of equations per time-step. Though more stable, but it can also become computationally expensive.

Apart from looking into their basic formulation this project will also study their stability properties, accuracy and computational load. Stability ensures small numerical errors do not grow exponentially over large times while accuracy depends on how good a numerical scheme captures the phenomena under investigation as modelled by partial differential equations. Trade-offs between these factors may occur especially when applied to solving the heat-schema where diffusion processes causes sometimes sudden changes in temperatures' distribution or in case compared literature-data are available.

Also, this approach for solving PDEs in grid form is not restricted to the simplest case; it can easily handle a variety of boundary conditions, which are conditions applied on the boundaries (that could be CFL or Neumann) that fix values or fix fluxes at the domain boundary. Merely by changing the distance between the grid points (grid spacing), changing in timestep setting (time steps between updates to compute difference expressions), and applying different boundary condition specifications, we can cover a great variety of problems such as heat flow in solid material or spreading of contamination in a fluid.

The main objective of this project is to study numerically how well finite difference methods work as alternatives for approximating solutions of ordinary and partial differential equations when dealing with physical problems. We will implement and perform some specific numerical calculations based on both an explicit scheme and two implicit schemes for ordinary differential equations, namely backward Euler's method and Crank-Nicolson scheme one. Similarly, we will also take advantage from explicit first-order schemes for newly designed implicit finite-difference model adjustments formulae when reformulating systems with respect to heat equation, along with semi-Lagrangian central differencing strategy in second order hyperbolic transport equation.

BACKGROUND

NUMERICAL METHODS FOR INTEGRATION OF ORDINARY DIFFERENTIAL METHODS

Runge-Kutta (RK) methods are an iterative class of numerical techniques used to solve ordinary differential equations (ODEs). Numerical analysts and applied mathematicians favor these methods because high-accuracy can be attained by reducing the necessity of computing high-order derivatives. In addition, two types of Runge-Kutta schemes- second order RK (RK2) and fourth order RK (RK4)- provide a balance between computational effort and accuracy and thus are sufficient for most physical as well as mathematical models such as non-linear, linear or chaotic systems.

Overview of Runge-Kutta Methods

Runge-Kutta methods solve initial value problems for ODEs of the form:

$$\frac{dy}{dt} = f(t, y), y(t_0) = y_0$$

by estimating the value of y at successive time steps. RK methods differ from simpler methods, such as Euler's method, by making multiple intermediate calculations (slopes) within a time step to achieve more accurate results.

RK2 Method (Second-Order):

The RK2 method improves the basic Euler method by taking an additional slope estimate at the midpoint of the interval, resulting in a more accurate estimate of the solution at the next time step. The general form of RK2 is:

$$y_{n+1} = y_n + k_2 k_1 = hf(t_n, y_n) k_2 = hf(t_n + \frac{h}{2}, y_n + \frac{k_1}{2})$$

Here, k_1 is the slope at the beginning of the interval, and k_2 is the slope at the midpoint. The use of this intermediate step provides improved accuracy compared to Euler's method.

RK4 Method (Fourth-Order):

The RK4 method is one of the most popular due to its accuracy and reasonable computational cost. It takes four different slope estimates within the interval and combines them to calculate the next value:

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

Where:

$$k_1 = hf(t_n, y_n)$$

$$k_2 = hf\left(t_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right)$$

$$k_3 = hf(t_n + \frac{h}{2}, y_n + \frac{k_2}{2})$$

$$k_4 = hf(t_n + h, y_n + k_3)$$

This method estimates the slopes at the beginning, midpoint, and end of the interval, leading to highly accurate results, even for complex systems.

Applications to Different Systems

In this project, both RK2 and RK4 methods were applied to three distinct systems: the spring-mass damper (linear) system, the predator-prey (non-linear) model, and the Lorenz system (chaotic). These systems illustrate the versatility of Runge-Kutta methods in handling a variety of dynamic behaviours.

1. **Spring-Mass Damper System (Linear System)**
   The spring-mass damper system is a classical linear system governed by second-order ODEs:

$$m\frac{d^2x}{dt^2} + c\frac{dx}{dt} + kx = F(t)$$

where m is the mass, c is the damping coefficient, k is the spring constant, and F(t) is the external force. We have converted this into a set of first-order ordinary differential equations (ODEs), and we have used the Runge-Kutta methods RK2 and RK4 for numerical integration to plot the oscillatory behavior of the system over a temporal framework. Both approaches capture the damped harmonic motion, as shown in figure 1a below. The time step has been set to 0.1 in order to emphasize the difference between the two methods.

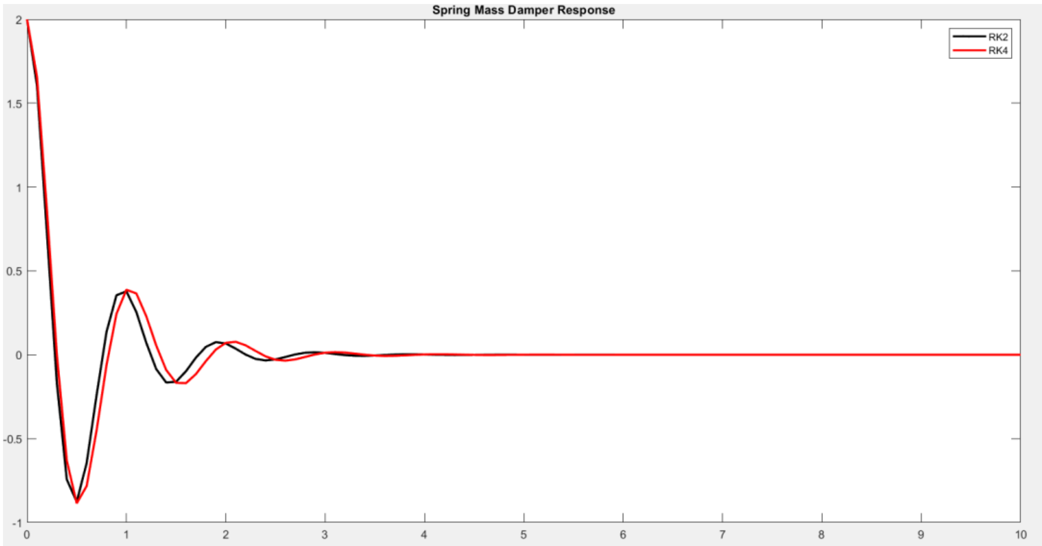


figure 1a: Spring Mass Damper Response

2. **Predator-Prey Model (Non-Linear System)**
The predator-prey system is a classic example of a non-linear system modelled by the Lotka-Volterra equations:

$$\frac{dx}{dt} = \alpha x - \beta xy$$

$$\frac{dy}{dt} = \delta xy - \gamma y$$

Here, x is the prey population, y is the predator population, and α, β, δ, γ are positive constants representing the interaction rates. The use of RK2 and RK4 enables one to track the oscillatory nature of both populations over time. This method is especially useful for this system since it controls the fast, nonlinear interactions between the predator and the prey much more effectively, thereby ensuring that the oscillations and the population dynamics are captured with an accuracy that would even for longer simulation times. Figure 1b below depicts the RK2 and RK4 differences:.

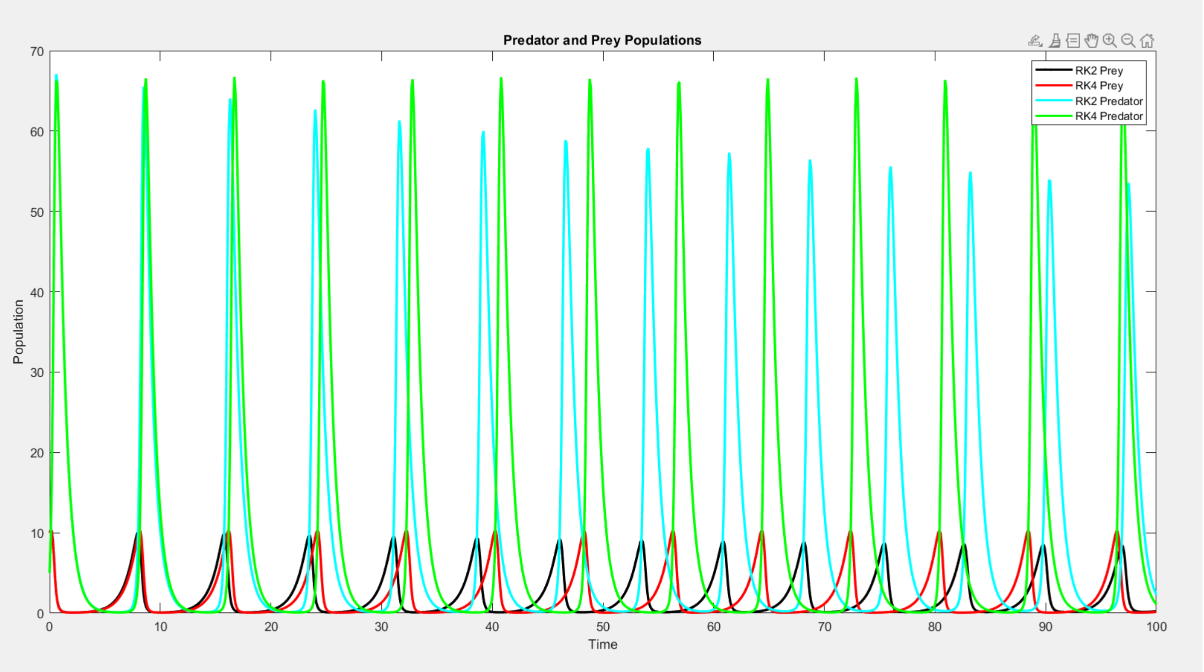


figure 1b: Predator and Prey Populations

3. **Lorenz System (Chaotic System)**
The Lorenz system, governed by the following set of non-linear differential equations, exhibits chaotic behaviour:

$$\frac{dx}{dt} = \sigma(y - x)$$

$$\frac{dy}{dt} = x(\rho - z) - y$$

$$\frac{dz}{dt} = xy - \beta z$$

In this scenario, σ, ρ, and β are parameters which are given as σ=10, ρ=28, and β=8/3 that are critical for the appearance of chaos; the Lorenz system is an example of a sensitive dynamical system which means that small changes in the initial conditions yield drastically different paths with time. The use of RK4 in this system provides a consistent way to numerically integrate the extremely sensitive equations involved in the system. While the Euler method may approximate the early stages of motion, RK4 is better suited to maintain stability and accuracy over longer time intervals and therefore more suitable for studying chaotic systems like the Lorenz attractor.

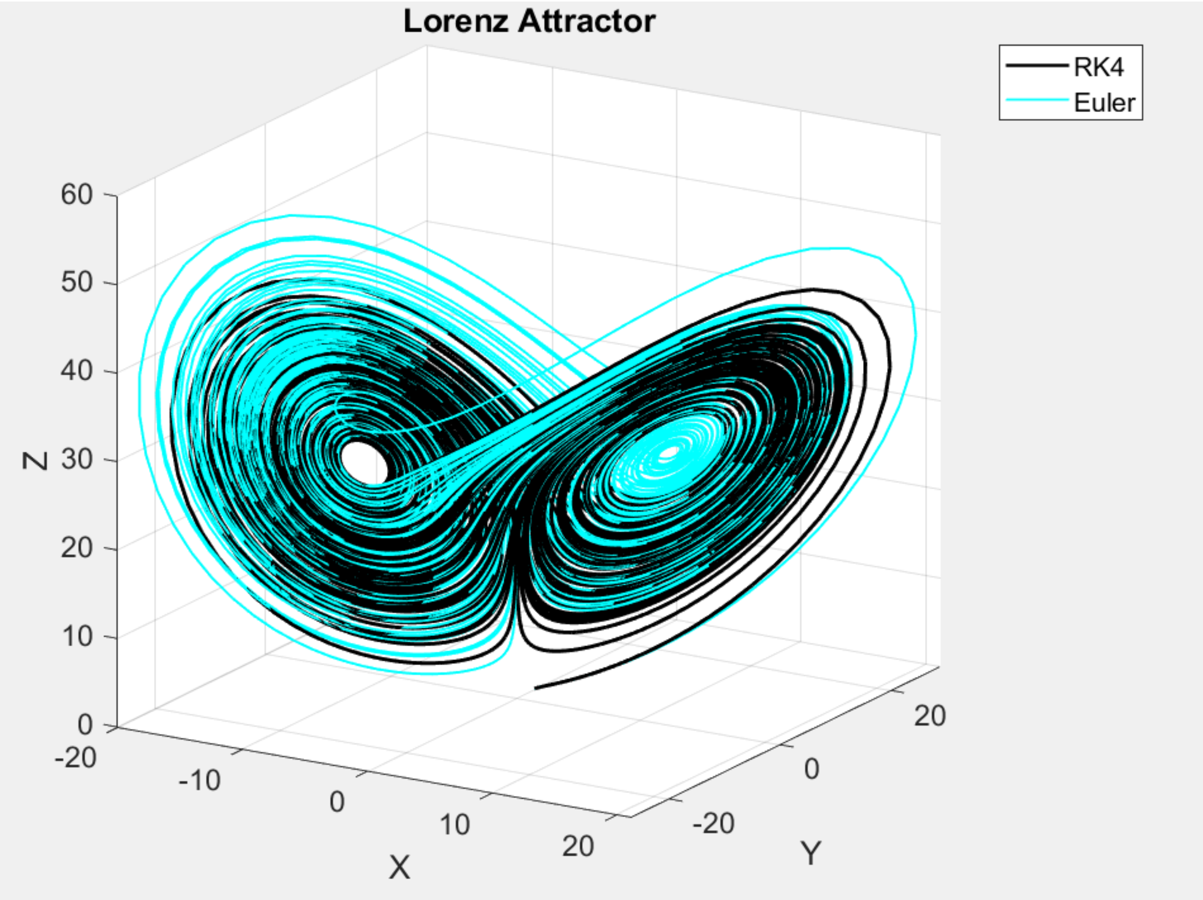


figure 1c: Lorenz Attractor

Conclusion

The use of Runge-Kutta methods, particularly RK2 and RK4, shows their versatility and efficiency in numerically solving ODEs across a wide spectrum of systems. RK2 provides a balance between simplicity and accuracy, making it suitable for less complex systems, while RK4 offers a higher degree of precision, especially for non-linear and chaotic systems.

INTRODUCTION TO FINITE DIFFERENCE SCHEMES TO PARTIAL DIFFERENCE EQUATIONS

1. FTFS (Forward in Time, Forward in Space)

The FTFS scheme is a simple finite difference method used to solve hyperbolic partial differential equations, such as the linear advection equation:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0$$

Here, u(x,t) is the quantity being advected, and c is the wave speed. The FTFS scheme approximates both the time and spatial derivatives using forward differences.

Time discretization (Forward in Time):

$$\frac{\partial u}{\partial t} \approx \frac{u_k^{n+1} - u_k^n}{\Delta t}$$

Space discretization (Forward in Space):

$$\frac{\partial u}{\partial x} \approx \frac{u_{k+1}^n - u_k^n}{\Delta x},$$

By substituting these into the original equation, the FTFS update formula is:

$$u_k^{n+1} = u_k^n - \lambda(u_{k+1}^n - u_k^n)$$

where:

u_k^{n+1} is the value of u at position k and time step n

Δt and Δx are the time and space step sizes, respectively,

$\lambda = \frac{c\Delta t}{\Delta x}$ is the Courant number (CFL condition).

This method is conditionally stable and can be unstable if the CFL condition is not satisfied, which is typically:

$$\frac{c\Delta t}{\Delta x} \leq 1$$

2. FTBS (Forward in Time, Backward in Space)

The FTBS scheme is another finite difference method used to solve hyperbolic partial differential equations like the linear advection equation:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0$$

Here, u(x,t) is the quantity being advected, and c is the wave speed. In the FTBS scheme, the time derivative is approximated using a forward difference, while the spatial derivative is approximated using a backward difference.

Time discretization (Forward in Time):

$$\frac{\partial u}{\partial t} \approx \frac{u_k^{n+1} - u_k^n}{\Delta t}$$

Space discretization (Backward in Space):

$$\frac{\partial u}{\partial x} \approx \frac{u_k^n - u_{k-1}^n}{\Delta x}$$

Substituting these into the advection equation gives the FTBS update formula:

$$u_k^{n+1} = u_k^n - \lambda(u_k^n - u_{k-1}^n)$$

where:

u_k^{n+1} is the value of u at position k and time step n

Δt and Δx are the time and space step sizes, respectively,

$\lambda = \frac{c\Delta t}{\Delta x}$ is the Courant number (CFL condition).

The FTBS scheme is conditionally stable and is generally more stable than the FTFS scheme. The stability criterion requires the CFL condition to be satisfied, which is:

$$\frac{c\Delta t}{\Delta x} \leq 1$$

This ensures that the numerical solution does not become unstable as time progresses.

3. FTCS (Forward in Time, Centered in Space)

The FTCS (Forward in Time, Centered in Space) scheme is a finite difference method used to solve the linear advection equation:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0$$

Here, u(x,t) represents the advected quantity, and c is the constant wave speed. In the FTCS scheme, the time derivative is approximated using a forward difference, while the spatial derivative is approximated using a centered difference.

Time discretization (Forward in Time):

$$\frac{\partial u}{\partial t} \approx \frac{u_k^{n+1} - u_k^n}{\Delta t}$$

Space discretization (Centered in Space):

$$\frac{\partial u}{\partial x} \approx \frac{u_{k+1}^n - u_{k-1}^n}{2\Delta x}$$

By substituting these into the advection equation, the FTCS update formula is:

$$u_k^{n+1} = u_k^n - (\lambda/2) \ (u_{k+1}^n - u_{k-1}^n)$$

where:

u_k^{n+1} is the value of u at position k and time step n

Δt and Δx are the time and space step sizes, respectively,

$\lambda = \frac{c\Delta t}{\Delta x}$ is the Courant number (CFL condition).

Stability:

Unlike the FTFS and FTBS schemes, the FTCS scheme is conditionally unstable for most hyperbolic problems like advection. Even though it provides second-order accuracy in space, its use for advection equations is generally avoided due to instability unless artificial damping or other stabilization methods are applied. The instability arises because there is no suitable CFL condition to prevent growing oscillations in the solution.

CONVERGENCE

The **convergence** of a Partial Differential Equation (PDE) to a finite difference scheme refers to the idea that the numerical solution obtained using the finite difference method approaches the exact solution of the PDE as the step sizes in both time and space (denoted as Δt and Δx respectively) become smaller.

Definition A difference scheme $L_k^n u_k^n = G_k^n$ approximating the partial differential equation Lv=F is a pointwise convergent scheme if for any x and t, as $(k\Delta x,(n+1)\Delta t)$ converges to (x,t), u_k^n converges to v(x,t) as Δx and Δt converge to 0.

Definition A difference scheme $L_k^n u_k^n = G_k^n$ approximating the partial differential equation Lv=F is a pointwise convergent scheme at time t if, as $(n+1)\Delta t \rightarrow t$, $\| u^{n+1} - v^{n+1}\| \rightarrow 0$ as $\Delta x \rightarrow 0$ and $\Delta t \rightarrow 0$

CONSISTENCY

Consistency in the context of finite difference schemes for solving Partial Differential Equations (PDEs) refers to how well the finite difference approximation represents the original PDE. A finite difference scheme is **consistent** with a PDE if, as the time step Δt and spatial step Δx approach zero, the difference between the finite difference approximation and the original PDE vanishes. In other words, the discretized equations must correctly approximate the continuous derivatives as the grid is refined.

Definition The finite difference scheme $L_k^n u_k^n = G_k^n$ is pointwise consistent with the partial differential equation Lv=F at point (x,t) if for any smooth function $\phi = \phi(x,t)$,

$$(L\phi - F)|_k^n - [L_k^n \phi(k\Delta x, n\Delta t) - G_k^n] \rightarrow 0$$

as $\Delta x, \Delta t \rightarrow 0$ and $(k\Delta x,(n+1)\Delta t) \rightarrow (x,t)$

choose ϕ to be the solution, *v*, to the partial differential equation. Then the expression in becomes $L_k^n u_k^n - G_k^n \rightarrow 0$ as Δx, $\Delta t \rightarrow 0$

If we write the difference scheme as (assuming now that we are working with a two-level scheme and a partial differential equation that is first order with respect to *t*)

$$u^{n+1} = Qu^n + \Delta t G^n$$

where

$$u^n = (\cdots, u_{-1}^n, u_0^n, u_1^n, \cdots)^T,$$

$$G^n = (\cdots, G_{-1}^n, G_0^n, G_1^n, \cdots)^T$$

and *Q* is an operator acting on the appropriate space, then a stronger definition of consistency can be given.

Definition The difference scheme $u^{n+1} = Qu^n + \Delta t G^n$ is consistent with the partial differential equation in a norm $\|.\|$ if the solution of the partial differential equation, *v*, satisfies

$$v^{n+1} = Qv^n + \Delta t G^n + \Delta t T^n ,$$

and $$\|T^n\| \rightarrow 0$$

as Δx , $\Delta t \rightarrow 0$, where v^n denotes the vector whose k-th component is $v(k\Delta x, n\Delta t)$

STABILITY

When solving PDEs using finite difference methods, stability refers to the requirement that small perturbations or errors (such as round-off or discretization errors) do not grow uncontrollably as the computation proceeds. This is crucial for obtaining accurate and reliable solutions.

Definition The difference scheme $u^{n+1} = Qu^n$, $n \geq 0$, is said to be stable with respect to the norm $\|.\|$ if there exist positive constants Δx_0 and Δt_0, and non-negative constants K and β so that

$$\| u^{n+1} \| \leq Ke^{t\boldsymbol{\beta}} \|u^0\|$$

for $0 \leq t = (n+1)\Delta t$, $0 < \Delta x \leq \Delta x_0$ and $0 < \Delta t \leq \Delta t_0$

Another characterization of stability that is often useful can be stated as:

The difference scheme $u^{n+1} = Qu^n$, $n \geq 0$, is said to be stable with respect to the norm $\|.\|$ if and only if there exist positive constants Δx_0 and Δt_0, and non-negative constants K and β so that

$$\| Q^{n+1} \| \leq Ke^{t\boldsymbol{\beta}}$$

for $0 \leq t = (n+1)\Delta t$, $0 < \Delta x \leq \Delta x_0$ and $0 < \Delta t \leq \Delta t_0$

THE LAX THEOREM

Lax Equivalence Theorem: A consistent, two level difference scheme for a well-posed linear initial-value problem is convergent if and only if it is stable.

Thus as long as we have a consistent scheme, convergence is synonymous with stability.

Lax Theorem : If a two-level difference scheme $u^{n+1} = Qu^n + \Delta t G^n$ is accurate of order (p, q) in the norm $\|.\|$ to a well-posed linear initial-value problem and is stable with respect to the norm $\|.\|$, then it is convergent of order (p, q) with respect to the norm $\|.\|$

One of the hypotheses in both the Lax Equivalence Theorem and the Lax Theorem is that the initial-value problem be well-posed. An initial-value problem is well-posed if it depends continuously upon its initial conditions.

APPLICATION OF FINITE DIFFERENCE SCHEMES TO PARTIAL DIFFERENTIAL EQUATIONS

1) THE TRANSPORT EQUATION

The **transport equation** is a fundamental partial differential equation (PDE) that models the movement of particles, substances, or heat through a medium. Its simplest form in one dimension is:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0$$

where u(x, t) represents the quantity being transported (such as a pollutant or fluid), and c is the constant velocity at which it moves through space. This equation describes how the distribution of u changes over time as it is transported along the x-axis with velocity c.

FTFS (Forward in Time, Forward in Space)

To solve the transport equation numerically, we first applied the **Forward-Time Forward-Space (FTFS)** scheme. This explicit method approximates the temporal and spatial derivatives using forward differencing:

$$\frac{\partial u}{\partial t} \approx \frac{u_k^{n+1} - u_k^n}{\Delta t}, \qquad \frac{\partial u}{\partial x} \approx \frac{u_{k+1}^n - u_k^n}{\Delta x},$$

Substituting these approximations into the transport equation yields the FTFS update formula:

$$u_k^{n+1} = u_k^n - \lambda(u_{k+1}^n - u_k^n)$$

Where $\lambda = \frac{c\Delta t}{\Delta x}$, also known as the Courant number.

While this method is easy to implement, it is known to be **conditionally unstable**. The scheme breaks down leading to oscillations and non-physical results, as seen in the figure 2a.

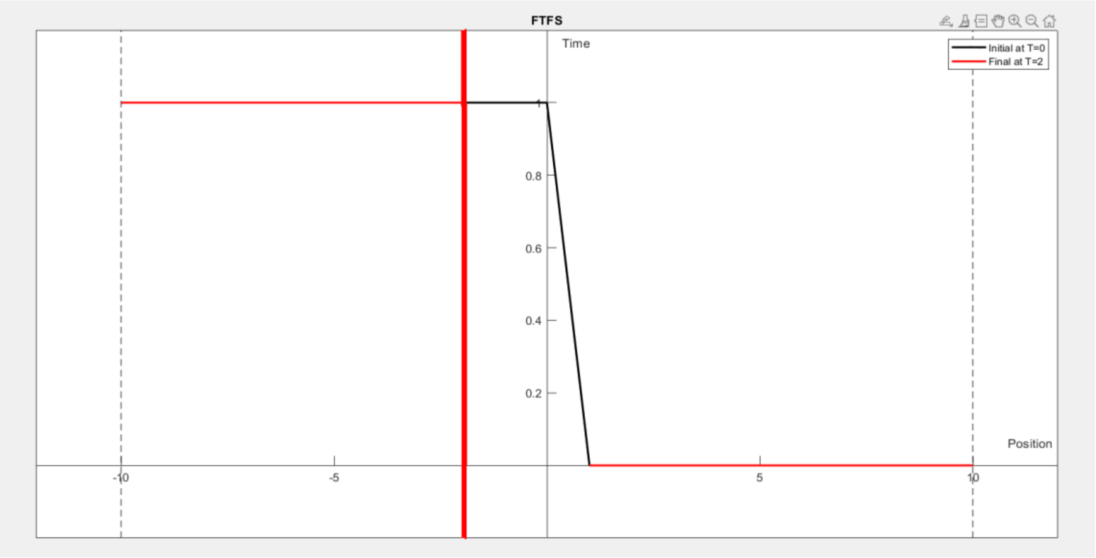


figure 2a: Transport Equation implemented with the FTFS scheme

FTBS (Forward in Time, Backward in Space)

To address the instability of FTFS, we then applied the **Forward-Time Backward-Space (FTBS)** scheme. In FTBS, the time derivative is again approximated with forward differencing, but the spatial derivative is now approximated with backward differencing:

$$\frac{\partial u}{\partial x} \approx \frac{u_k^n - u_{k-1}^n}{\Delta x}$$

The FTBS update formula becomes:

$$u_k^{n+1} = u_k^n - \lambda(u_k^n - u_{k-1}^n)$$

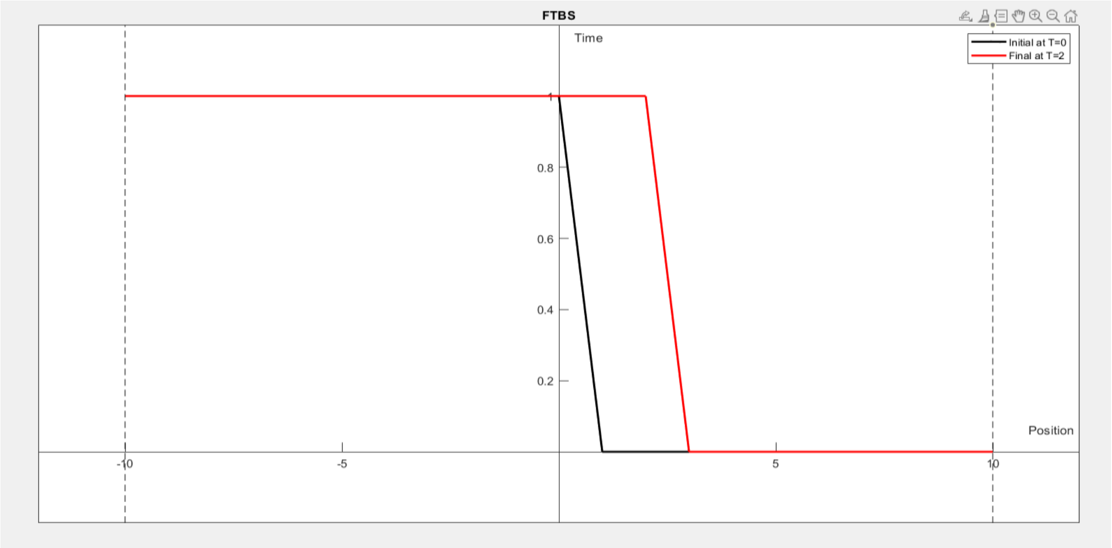


figure 2b: Transport Equation implemented with the FTBS scheme

This scheme is generally **more stable** than FTFS, even for larger Courant numbers, as observed in our results. As seen in figure 2b and figure 2c, the FTBS method successfully provided a stable approximation of the transport equation, even under conditions where FTFS failed.

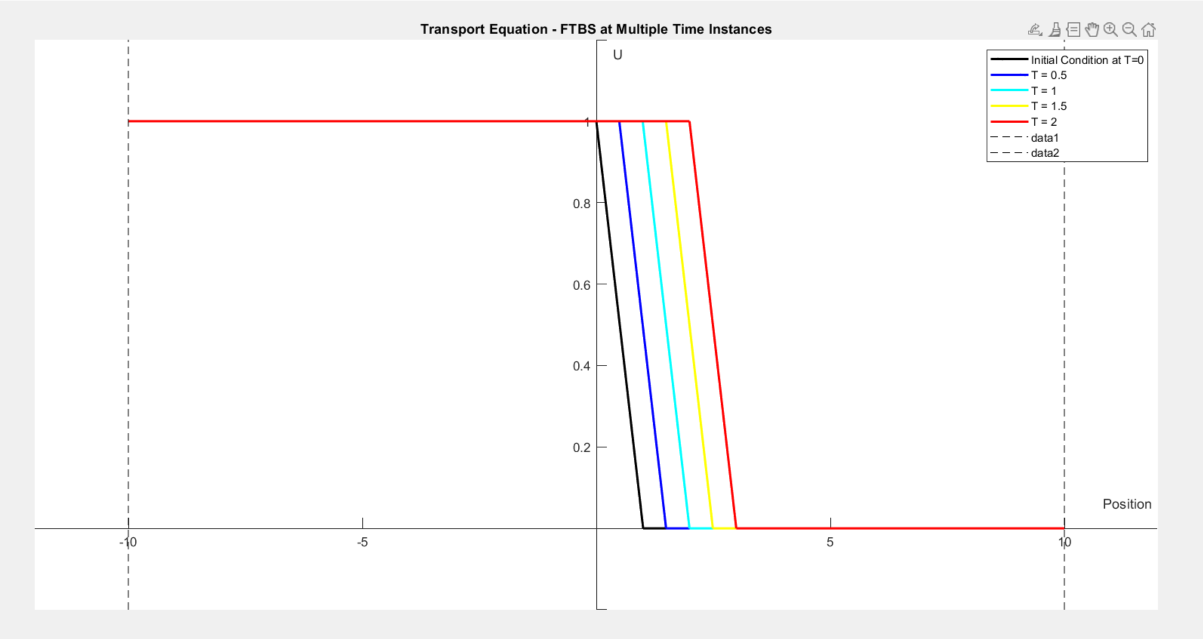


figure 2c: Transport Equation implemented with the FTBS scheme at multiple time instances

2) THE HEAT EQUATION

The heat equation is a partial differential equation that models the distribution of heat (or temperature) in a given region over time. It is widely used in fields such as physics, engineering, and material science to describe how heat diffuses through a medium. In one dimension, the heat equation is written as:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$$

where:

- u(x, t) is the temperature at position x and time t.
- α is the thermal diffusivity constant, which describes how quickly heat spreads through the material.

The heat equation describes the change in temperature with respect to time as a function of the spatial variation of temperature. Numerical methods like the Finite Difference Method are often used to approximate solutions to the heat equation when analytical solutions are not feasible, especially in complex geometries or with variable boundary conditions.

FTCS (Forward in Time, Centred in Space)

The **Forward-Time Central-Space (FTCS)** scheme is a numerical method used to solve the heat equation. In this explicit scheme, the time derivative is approximated using a forward difference, and the second-order spatial derivative is approximated using a central difference:

$$\frac{\partial u}{\partial t} \approx \frac{u_k^{n+1} - u_k^n}{\Delta t}, \qquad \frac{\partial^2 u}{\partial x^2} \approx \frac{u_{k+1}^n - 2u_k^n + u_{k-1}^n}{\Delta x^2}$$

Substituting these approximations into the heat equation gives the FTCS update formula:

$$u_k^{n+1} = u_k^n - \lambda(u_{k+1}^n - 2u_k^n + u_{k-1}^n)$$

Where $\lambda = \alpha \frac{\Delta t}{\Delta x^2}$

In our project, we implemented the FTCS scheme to model heat diffusion across a rod over time. The initial condition was set as a sinusoidal temperature distribution, and boundary conditions were applied to ensure the temperature at both ends of the rod remained fixed at zero. The time evolution of the temperature distribution was computed for various time steps and spatial resolutions.

The FTCS scheme allows us to simulate the diffusion of heat by iterating the solution over multiple time steps. However, the stability of the method is conditional, requiring $\lambda \leq 0.5$ to avoid instability. This constraint means that small time steps must be chosen relative to the spatial resolution to ensure accurate results.

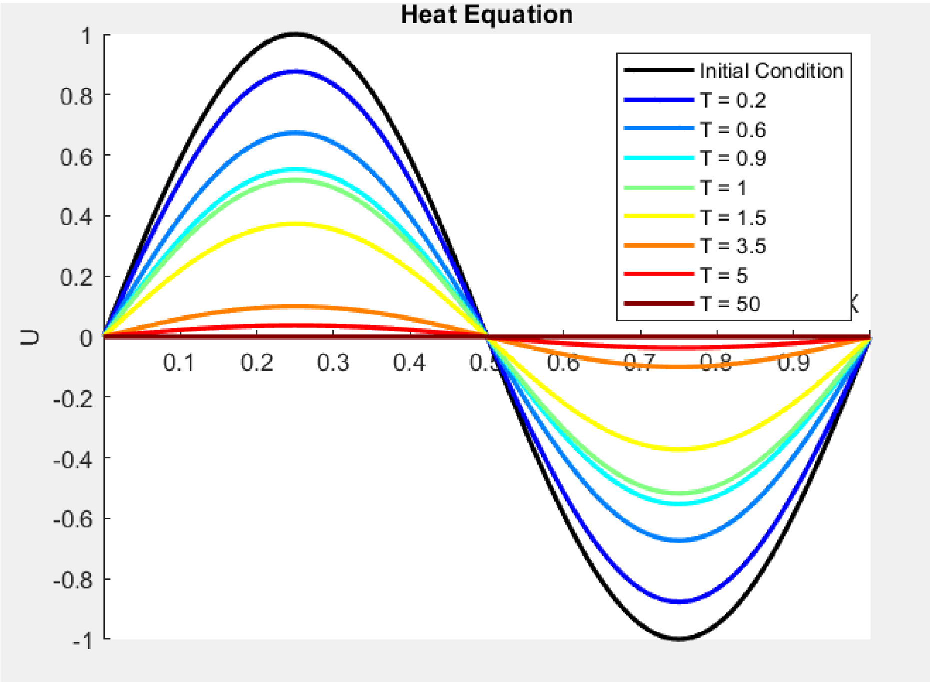


figure 3a: Heat Equation implemented with the FTCS scheme

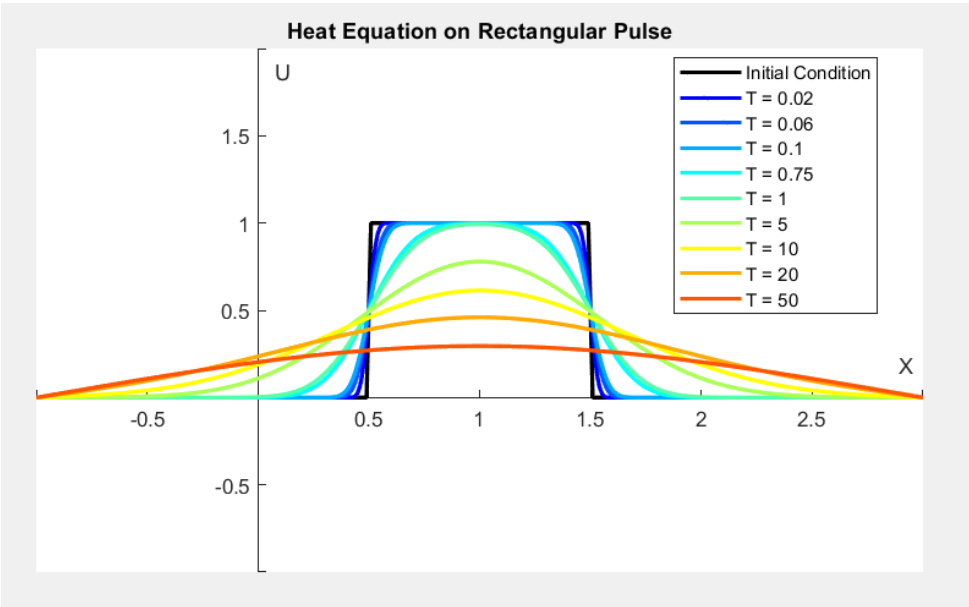


figure 3b: Heat Equation implemented with the FTCS scheme on rectangular pulse

COMPARISON WITH ANALYTICAL SOLUTION

Figure 3c compares the performance of the FTCS (Forward-Time Central-Space) scheme used to solve the heat equation against its known analytical solution. The initial condition, a sharp waveform, is shown in black. The numerical solution, obtained using the FTCS method, is displayed in red, while the green curve represents the exact analytical solution. By increasing the time step Δt we can observe a clear difference between the numerical and analytical results, demonstrating how stability and accuracy of the numerical scheme are affected. This difference becomes more pronounced as the time step increases, as larger Δt leads to increased error in the numerical integration, deviating from the analytical solution.

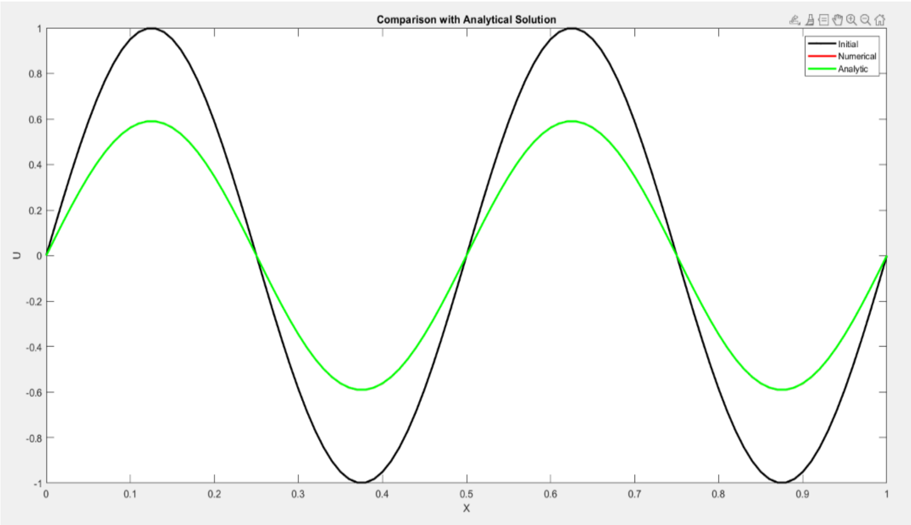


figure 3c: Comparison of Numerical and Analytical solutions of Heat Equation implemented with the FTCS scheme

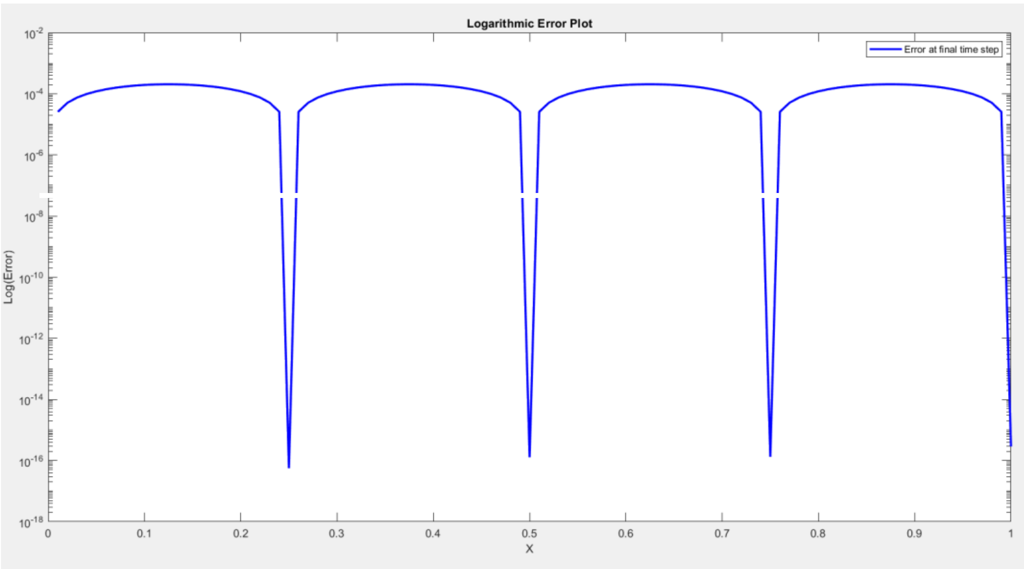


figure 3d: Logarithmic error plot of Heat Equation implemented with the FTCS scheme

VON NEUMANN STABILITY ANALYSIS OF FINITE DIFFERENCE SCHEMES

The Von Neumann stability analysis is a technique employed to assess the stability of numerical methods when solving partial differential equations, especially in the context of the transport (advection) equation. In this analysis, we focus on the FTFS (Forward-Time Forward-Space) and FTBS (Forward-Time Backward-Space) schemes as they pertain to the transport equation, and the FTCS (Forward-Time Centered-Space) scheme as applied to both the transport and heat equation.

FTFS Scheme:

Previously we have represented the FTFS schemes as applied to the transport equation as follows:

$$u_j^{n+1} = u_j^n - \lambda\left(u_{j+1}^n - u_j^n\right)\_\_\_\_\_\_\_\_\_ (1)$$

Where $\lambda = \frac{c\Delta t}{\Delta x}$ also known as the Courant number.

We compute the Fourier mode solution by substituting u_j^n as follows:

$$u_j^n = \xi^n e^{ikj\Delta x} \_\_\_\_\_\_\_\_\_(2)$$

Where k is the wave number, j is the spatial index, n is the temporal index, Δx is the and ξ is called the amplification factor.

If $|\xi| > 1$ for any k, then the scheme is deemed unstable.

Substituting (2) in (1):

$$\xi^{(n+1)} e^{ikj\Delta x} = \xi^n e^{ikj\Delta x} - \lambda(\xi^n e^{ik(j+1)\Delta x} - \xi^n e^{ikj\Delta x})$$

On simplification:

$$\xi = 1 - \lambda\left(e^{ikj\Delta x} - 1\right)\_\_\_\_\_\_\_\_(3)$$

Expanding $e^{ikj\Delta x}$ into polar form:

$$e^{ikj\Delta x} = cos\ cos\ (k\Delta x) + isin(k\Delta x)$$

Substituting back into (3)

$$\xi = 1 - \lambda(cos\ cos\ (k\Delta x) - 1) - i\lambda sin(k\Delta x)$$

$$|\xi| = \sqrt{(1 - \lambda(cos\ cos\ (k\Delta x) - 1))^2 + (\lambda sin(k\Delta x))^2}$$

$$|\xi|^2 = (1+\lambda)^2 - 2\lambda(1+\lambda)\ cos\ cos\ (k\Delta x) + \lambda^2$$

On analysing edge cases:

1) When $k\Delta x = 0$:

$$|\xi| = 1$$

When $k\Delta x = \pm\pi$:

$$|\xi| = |1 + 2\lambda| Therefore, keeping\ in\ mind\ that\ \lambda$$

2) must satisfy the condition $-1 \leq |1 + 2\lambda| \leq 1$ we conclude that for FTFS to be stable, λ must satisfy the following inequality: $-1 \leq \lambda \leq 0$. Where $\frac{\Delta t}{\Delta x} \leq 1$ and $c < 0$.

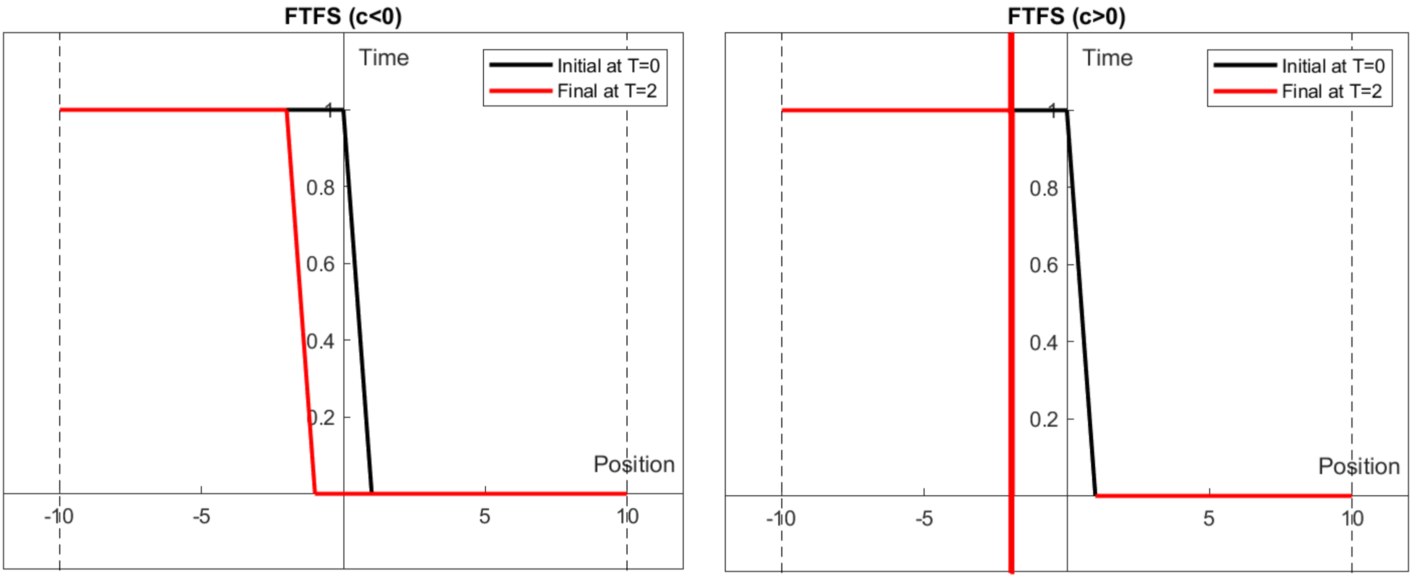


Figure 4a: Comparison of FTFS scheme applied to transport equation with conditions: (left) c=-1, (right) c=+1.

FTBS Scheme:

Previously we have represented the FTBS schemes as applied to the transport equation as follows:

$$u_j^{n+1} = u_j^n - \lambda\left(u_j^n - u_{j-1}^n\right) \_\_\_\_\_\_\_\_ (1)$$

Substituting u_j^n with Fourier mode solution:

$$\xi^{(n+1)} e^{ikj\Delta x} = \xi^n e^{ikj\Delta x} - \lambda(\xi^n e^{ikj\Delta x} - \xi^n e^{ik(j-1)\Delta x})$$

On simplification:

$$\xi = 1 - \lambda(1 - e^{-ikj\Delta x}) \_\_\_\_\_\_ (2)$$

Expanding $e^{-ikj\Delta x}$ into polar form:

$$e^{-ikj\Delta x} = cos\, cos\,(k\Delta x) - isin(k\Delta x)$$

Substituting back into (2)

$$\xi = 1 - \lambda\left((k\Delta x)\right) - i\lambda sin(k\Delta x)$$

$$|\xi| = \sqrt{\left(1 - \lambda\left((k\Delta x)\right)\right)^2 + \left(\lambda sin(k\Delta x)\right)^2}$$

$$|\xi|^2 = (1-\lambda)^2 - 2\lambda(1-\lambda)\, cos\, cos\,(k\Delta x) + \lambda^2$$

On analysing edge cases:

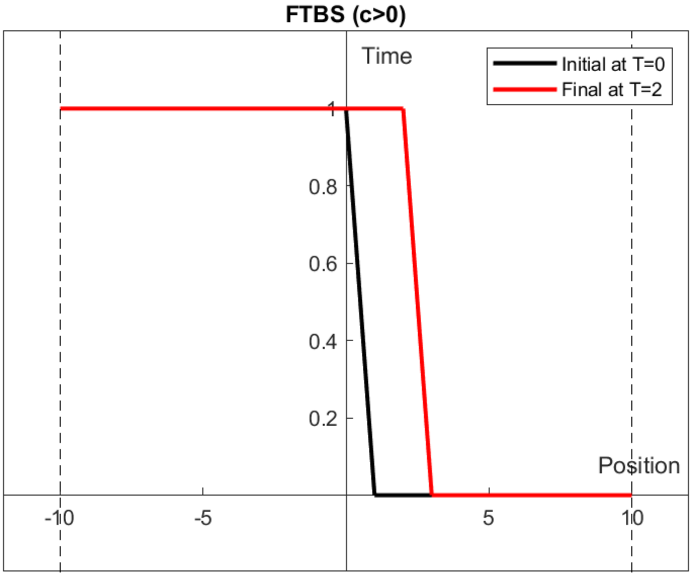
When $k\Delta x = 0$:

$$|\xi|^2 = (2\lambda - 1)^2$$

When $k\Delta x = \pm\pi$:

$$|\xi|^2 = 2\lambda(\lambda - 1)$$

To satisfy the condition $|\xi| \leq 1$ it is evident from both cases that λ must satisfy the inequality $0 \leq \lambda \leq 1$ where $c > 0$.



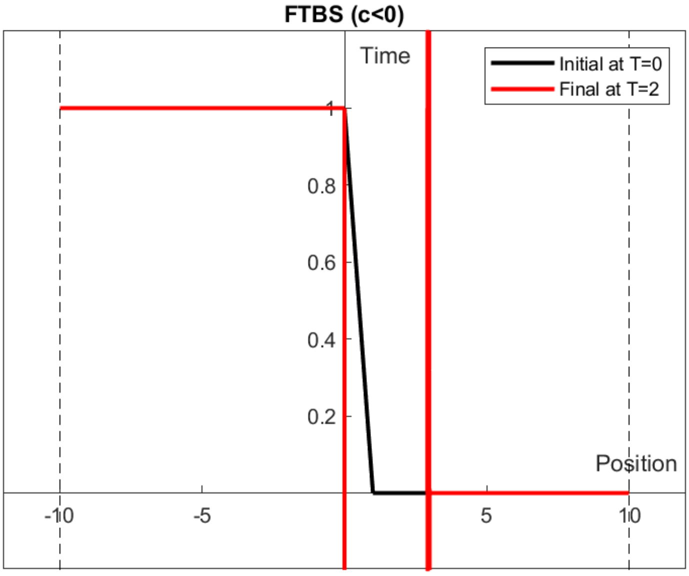


Figure 4b: Comparison of FTBS scheme applied to transport equation with conditions: and (left) c=+1, (right) c=-1.

Note on upwind schemes:

In the field of computational fluid dynamics, upwind schemes play a vital role in solving the transport equation numerically. Key examples of these schemes include Forward Time Forward Space (FTFS) and Forward Time Backward Space (FTBS). The fundamental idea revolves around the propagation of information in the transport equation along characteristics traveling at speed c.

To ensure stability in numerical schemes, it is essential to align with this physical flow of information:

- FTFS will be stable only when $c < 0$ (indicating a leftward flow).

- FTBS will maintain stability when $c > 0$ (indicating a rightward flow).

This understanding leads to what is known as the "upwind principle," which advocates for spatial discretizations that account for the direction of flow. The validity of upwind schemes in maintaining numerical stability is supported by von Neumann stability analysis, which demonstrates that these schemes are stable when they align with the actual direction of information flow. Additionally, the Courant-Friedrichs-Lewy (CFL) condition, expressed as $|\lambda| \leq 1$, is also necessary for ensuring stability.

FTCS Scheme:

We will be analysing the stability of the FTCS scheme in the context of both transport as well as heat equation to compare its distinct behaviour with both.

First let us apply it to the transport equation:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0$$

On discretization:

$$u_j^{n+1} = u_j^n - \frac{\lambda}{2}\left(u_{j+1}^n - u_{j-1}^n\right)$$

Where $\lambda = \frac{c\Delta t}{\Delta x}$.

Substituting u_j^n with Fourier mode solution:

$$\xi^{(n+1)}e^{ikj\Delta x} = \xi^n e^{ikj\Delta x} - \frac{\lambda}{2}(\xi^n e^{ik(j+1)\Delta x} - \xi^n e^{ik(j-1)\Delta x})$$

On simplification:

$$\xi = 1 - \frac{\lambda}{2}\left(e^{ikj\Delta x} - e^{-ikj\Delta x}\right)$$

$$= 1 - i\frac{\lambda}{2}sin(k\Delta x)$$

$$|\xi| = \sqrt{1 + (\frac{\lambda}{2}sin\,(k\Delta x))^2} \text{\_\_\_\_\_\_\_\_}(1)$$

It is evident from (1) that $|\xi|$ is always greater than 1. Therefore we conclude that FTCS scheme is unconditionally unstable when applied to the transport equation.

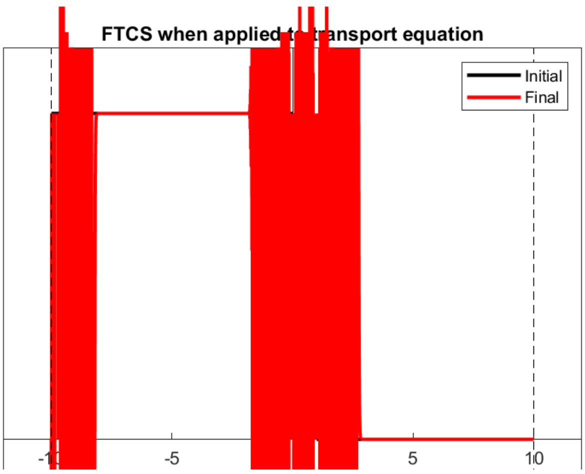


Figure 4c: FTCS is unconditionally unstable when applied to transport equation, irrespective of the value of chosen

Now let us apply the FTCS scheme to the heat equation:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$$

On discretization:

$$u_j^{n+1} = u_j^n + \lambda\left(u_{j+1}^n - 2u_j^n + u_{j-1}^n\right)$$

Where $\lambda = \frac{c\Delta t}{(\Delta x)^2}$.

Substituting u_j^n with Fourier mode solution:

$$\xi^{(n+1)} e^{ikj\Delta x} = \xi^n e^{ikj\Delta x} + \lambda(\xi^n e^{ik(j+1)\Delta x} - 2\xi^n e^{ikj\Delta x} + \xi^n e^{ik(j-1)\Delta x})$$

On simplification:

$$\xi = 1 - \lambda\left(e^{ikj\Delta x} - 2 + e^{-ikj\Delta x}\right)$$

$$= 1 + 2\lambda(cos\ cos\ (k\Delta x)\ - 1)$$

$$|\xi| = |1 + 2\lambda(cos\ cos\ (k\Delta x)\ - 1)|$$

On analysing edge cases:

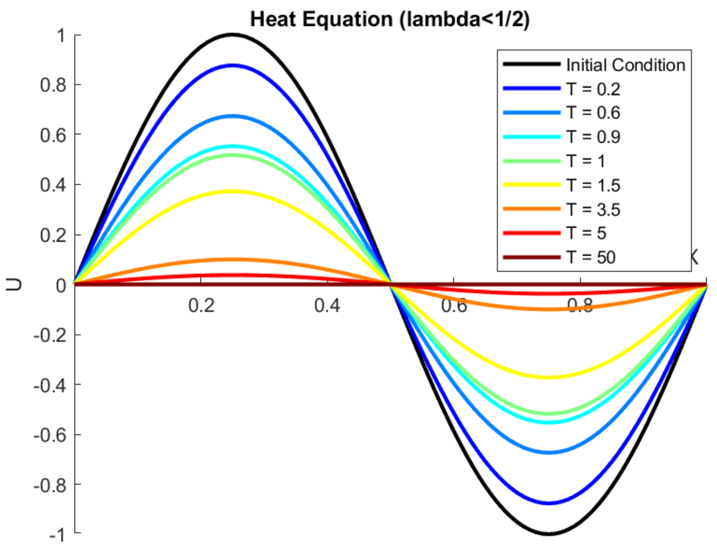
When $k\Delta x = 0$:

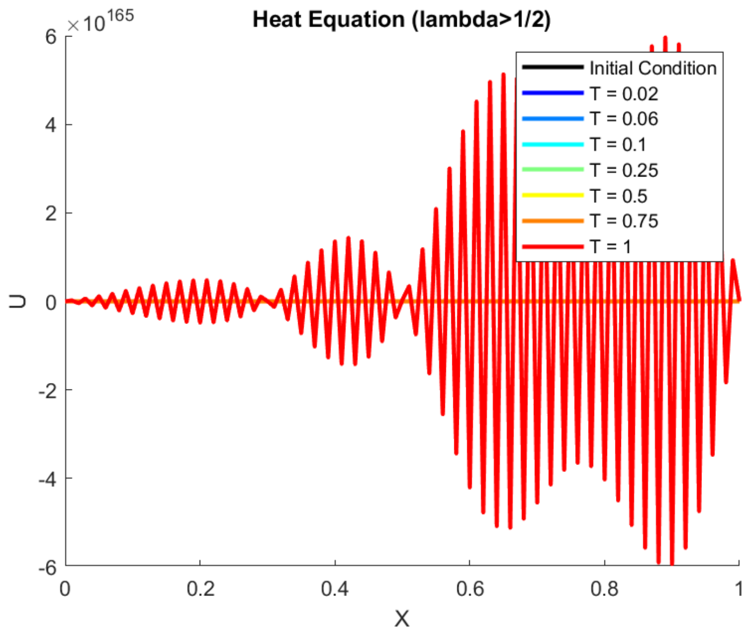
$$|\xi| = |2\lambda|$$

When $k\Delta x = \pm\pi$:

$$|\xi| = 0$$

We therefore conclude that to satisfy the condition $|\xi| \leq 1$, λ must satisfy the inequality $0 \leq \lambda \leq \frac{1}{2}$





Explicit Schemes (Lax - Wendroff Scheme)

1) The Transport Equation

To obtain a higher order stable scheme, we alter the centred scheme to stabilize it further.

For the Transport Equation,

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0$$

we see that $u_t = -cu_x$ and $u_{tt} = c^2 u_{xx}$.

Then by using Taylor series expansion,

$$u_k^{n+1} = u_k^n + u_{tk}^n \,\Delta t + u_{ttk}^n \,\Delta t^2/2 + O(\Delta t^3)$$

Substituting u_t , u_{tt} and applying the centred scheme we get,

$$u_k^{n+1} = u_k^n - R/2\ \delta_0\, u_k^n + R^2/2\ \delta^2\, u_k^n$$

where $R = c\Delta t/\Delta x$. This is called the **Lax Wendroff Scheme.** It is conditionally stable with the condition $|R| = |c|\,\Delta t/\Delta x \le 1$.

As seen in Figure 5a, when we take c = 1 and $|R| \le 1$, the scheme is stable for the Transport Equation.

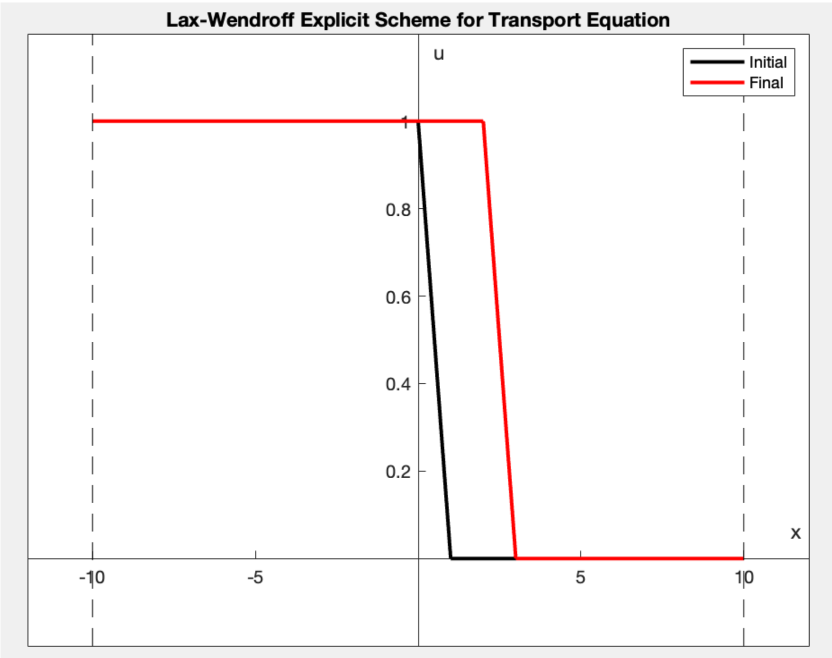


Figure 5a: Lax-Wendroff Scheme for Transport Equation

2) The Heat Equation

Deriving the lax Wendroff Scheme for the Heat Equation:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$$

we see that u_t = - cu_{xx} and u_{tt} = $\alpha^2 u_{xxxx}$

Then by using taylor series expansion, substituting u_t , u_{tt} and applying the centred scheme, we get the lax wendroff scheme.

Figure 5b compares the performance of the Lax Wendroff scheme used to solve the heat equation against its known analytical solution. The numerical solution, obtained using the Lax Wendroff method, is displayed in red, while the green curve represents the exact analytical solution. By increasing the time step Δt we can observe a clear difference between the numerical and analytical results, demonstrating how stability and accuracy of the numerical scheme are affected. This difference becomes more pronounced as the time step increases, as larger Δt leads to increased error in the numerical integration, deviating from the analytical solution.

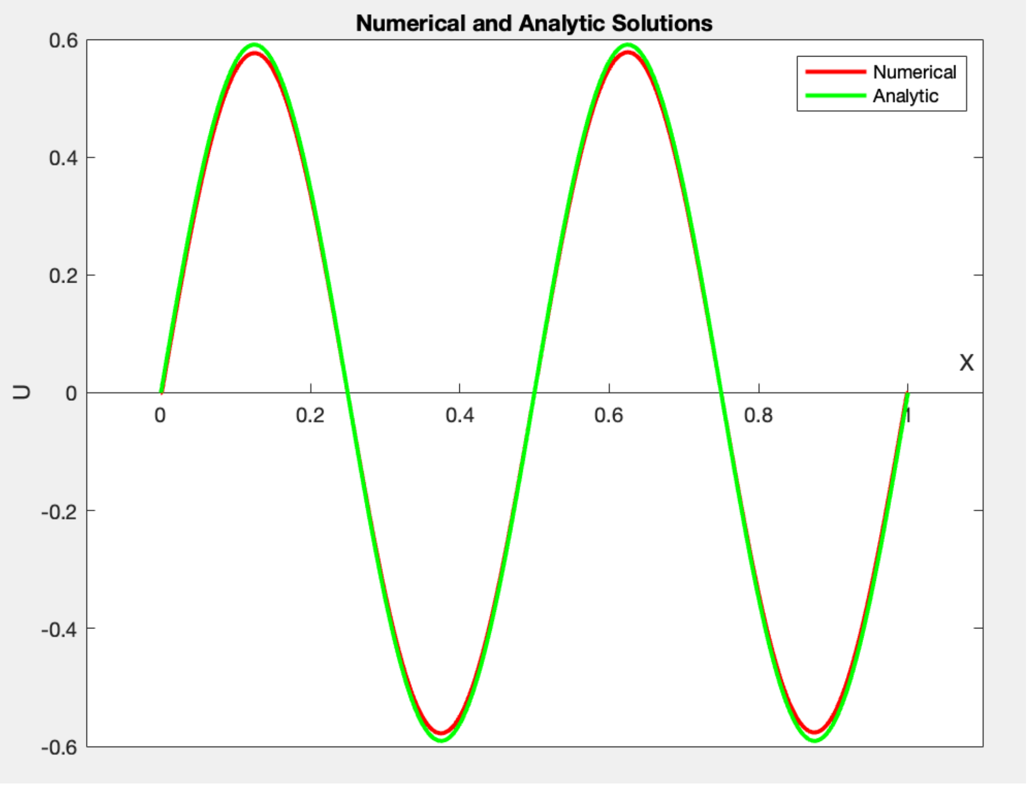


Figure 5b: Comparison of Numerical and Analytical solutions of Heat Equation implemented with the Lax Wendroff scheme

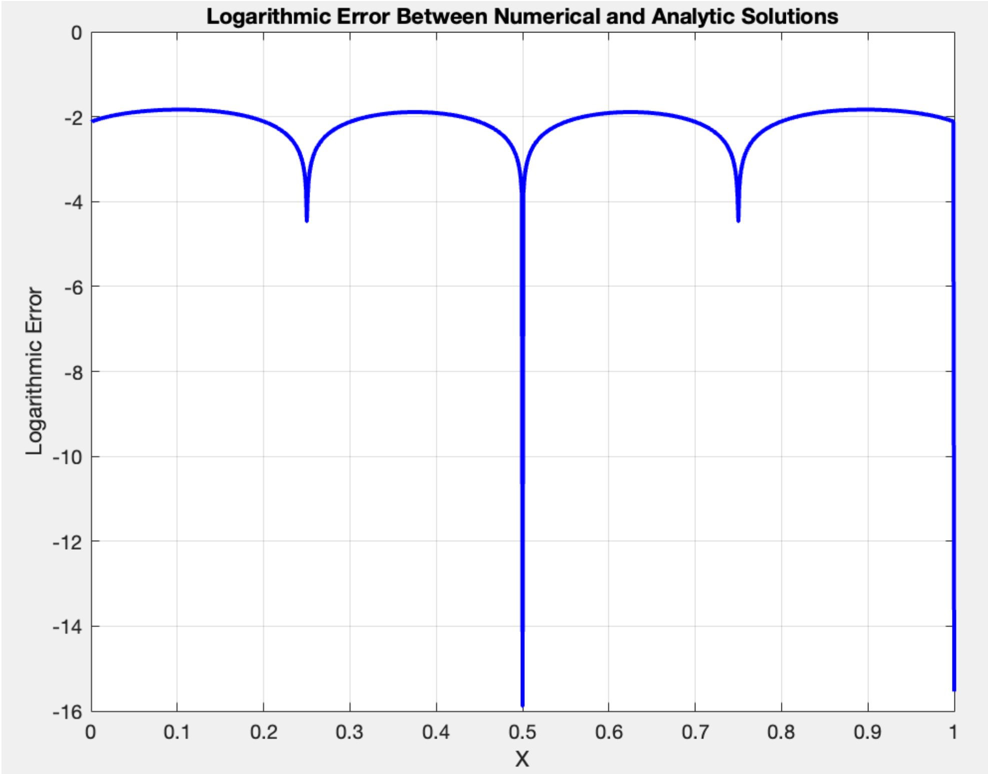


Figure 5c: Logarithmic error plot of Heat Equation implemented with the Lax Wendroff scheme

IMPLICIT SCHEMES

Implicit numerical methods use their current and succeeding time steps for each point. They have only one main disadvantage – unconditional stability. In other words, they do not depend on the time step size. In the scope of our project, we'll realize the following:

1. For the transport equation: Backward Time Forward Space (BTFS) and Backward Time Backward Space (BTBS) schemes

2. For the heat equation: Backward Time Central Space scheme (BTCS) scheme

However, these implicit schemes come with the cost of requiring to solve a system of equations for each step. Implicit schemes are indeed more expensive in terms of computation than explicit schemes but their stability benefits often makes the extra cost justifiable particularly in instances where large time steps are required.

BTFS (Backward in Time, Forward in Space)

The Backward Time Forward Space (BTFS) scheme uses backward differencing in time and forward differencing in space to solve the transport equation:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0$$

Time discretization:

$$\frac{\partial u}{\partial t} = \frac{u_k^{n+1} - u_k^n}{\Delta t}$$

Space discretization:

$$\frac{\partial u}{\partial x} = \frac{u_{k+1}^{n+1} - u_k^{n+1}}{\Delta x} Substituting\ into\ the\ transport\ equation: (1-\lambda)u_k^{n+1} + \lambda u_{k+1}^{n+1} = u_k^n \_\_\_\_\_\_ (1) Where\ \lambda = \frac{c\Delta t}{\Delta x}$$

.

Since the BTFS scheme involves u_{k+1}^{n+1} , which is unknown at the new time level, we must form and solve a linear system of equations at each time step to compute u_k^{n+1} for all spatial points.

Performing Von-Neumann stability analysis on (1):

$$\begin{bmatrix} 1-\lambda & \lambda & 0 & 0 & \cdots & 0 \\ 0 & 1-\lambda & \lambda & 0 & \cdots & 0 \\ 0 & 0 & 1-\lambda & \lambda & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1-\lambda & \lambda \\ 0 & 0 & 0 & \cdots & 0 & 1-\lambda \end{bmatrix} \begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_m^{n+1} \end{bmatrix} = \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ \vdots \\ u_m^n \end{bmatrix}$$

$$\xi = \frac{1}{1 - \lambda + \lambda e^{ik\Delta x}}$$

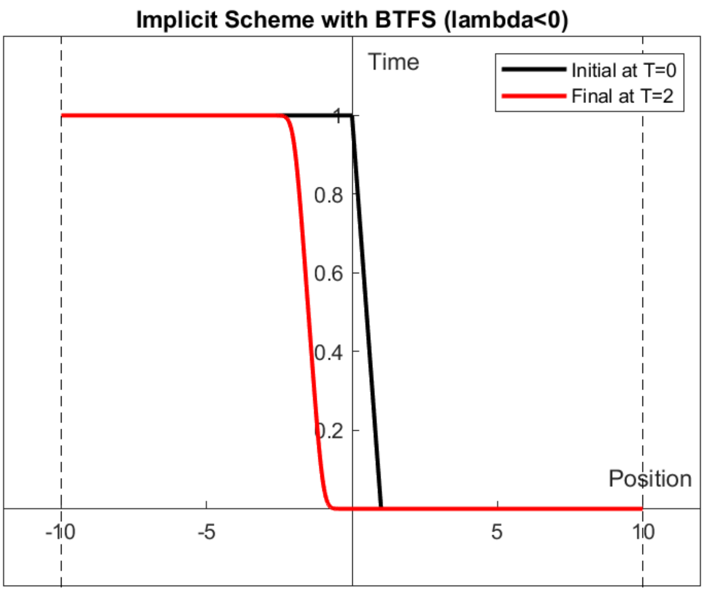
$$= \frac{1}{1 - \lambda + \lambda\, cos\, cos\, (k\Delta x)\ + i\lambda sin\, (k\Delta x)}$$

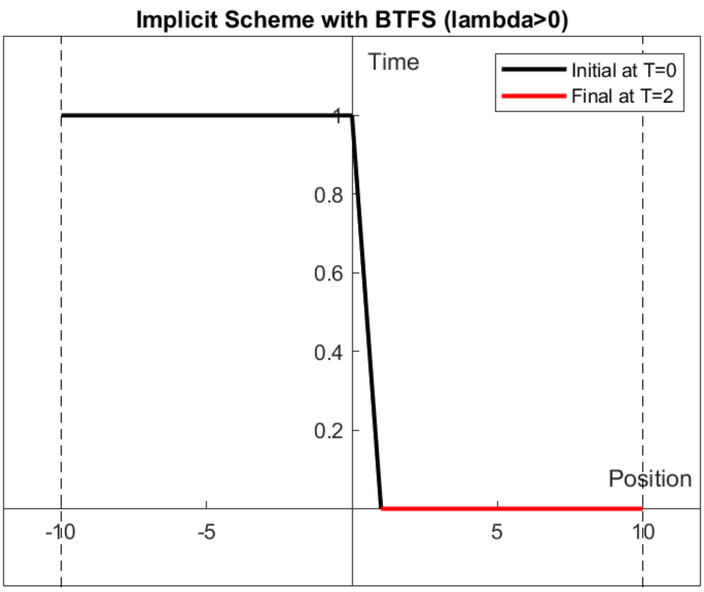
Taking modulus and simplifying:

$$|\xi|^2 = \frac{1}{1 - 4\lambda sin^2 \frac{k\Delta x}{2} + 4\lambda^2 sin^2 \frac{k\Delta x}{2}}$$

$$= \frac{1}{1 - 4\lambda sin^2 \frac{k\Delta x}{2} (1 - \lambda)} \_\_\_\_\_ (2)$$

On analyzing (2) we conclude that for $\lambda > 0, |\xi| \geq 1$. Therefore for BTFS to be stable on the transport equation, λ must satisfy the condition $\lambda \leq 0$.





BTBS (Backward in Time, Backward in Space)

The Backward Time Backward Space (BTBS) scheme applies backward differencing in both time and space for solving the transport equation.

Time discretization:

$$\frac{\partial u}{\partial t} = \frac{u_k^{n+1} - u_k^n}{\Delta t}$$

Space discretization:

$$\frac{\partial u}{\partial x} = \frac{u_k^{n+1} - u_{k-1}^{n+1}}{\Delta x} Substituting\ into\ the\ transport\ equation: -\lambda u_{k-1}^{n+1} + (1+\lambda)u_{k+1}^{n+1} = u_k^n \_\_\_\_\_\_(1) Where\ \lambda = \frac{c\Delta t}{\Delta x}$$

.

In order to solve the transport equation using the BTBS scheme, we need to establish a system of linear equations at each time step. The unknown values at the next time step, denoted as u_k^{n+1}, will be linked to the known values from the current time step, u_k^n, through the discretized equations.

$$\begin{bmatrix} 1+\lambda & 0 & 0 & \cdots & 0 & 0 \\ -\lambda & 1+\lambda & 0 & \cdots & 0 & 0 \\ 0 & -\lambda & 1+\lambda & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\lambda & 1+\lambda \end{bmatrix} \begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ u_m^{n+1} \end{bmatrix} = \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ \vdots \\ u_m^n \end{bmatrix}$$

Performing Von-Neumann stability analysis on (1):

$$\xi = \frac{1}{1 + \lambda - \lambda e^{-ik\Delta x}}$$

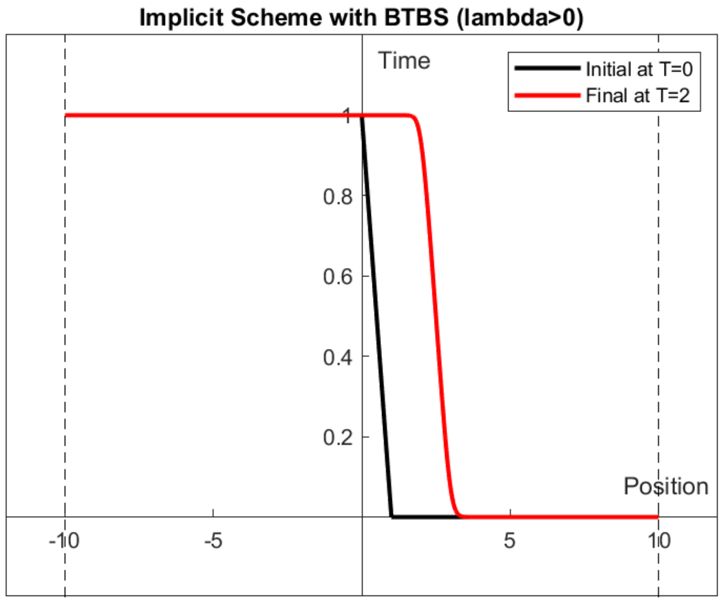
$$= \frac{1}{1 + \lambda - \lambda\, cos\, cos\, (k\Delta x)\ + i\lambda sin\, (k\Delta x)}$$

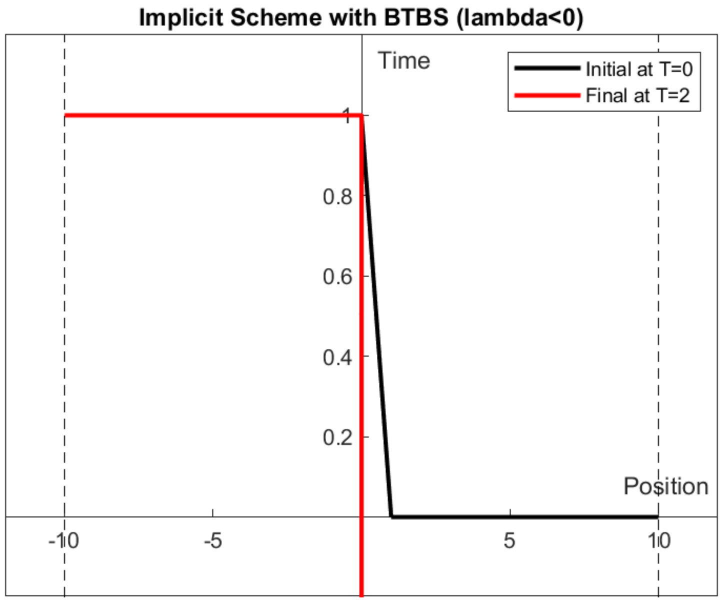
Taking modulus and simplifying:

$$|\xi|^2 = \frac{1}{1 + 2\lambda sin^2 \frac{k\Delta x}{2} + 2\lambda^2 sin^2 \frac{k\Delta x}{2}}$$

$$= \frac{1}{1 + 2\lambda sin^2 \frac{k\Delta x}{2} (1+\lambda)} \_\_\_\_\_(2)$$

On analyzing (2) we conclude that for $\lambda < 0, |\xi| \geq 1$. Therefore for BTFS to be stable on the transport equation, λ must satisfy the condition $\lambda \geq 0$.





BTCS (Backward in Time, Centered in Space)

The Backward Time Central Space (BTCS) scheme combines backward differencing in time with central differencing in space to solve the heat equation:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$$

Time discretization:

$$\frac{\partial u}{\partial t} = \frac{u_k^{n+1} - u_k^n}{\Delta t}$$

Space discretization:

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{k+1}^{n+1} - 2u_k^{n+1} + u_{k-1}^{n+1}}{(\Delta x)^2}$$

Substituting into heat equation:

$$-\lambda u_{k-1}^{n+1} + (1 + 2\lambda)u_k^{n+1} - \lambda u_{k+1}^{n+1} = u_k^n \_\_\_\_\_\_(1)$$

To solve the transport equation using the BTCS scheme, we need to establish a system of linear equations at each time step. The unknown values at the next time step, denoted as u_k^{n+1}, will be linked to the known values from the current time step, u_k^n, through the discretized equations.

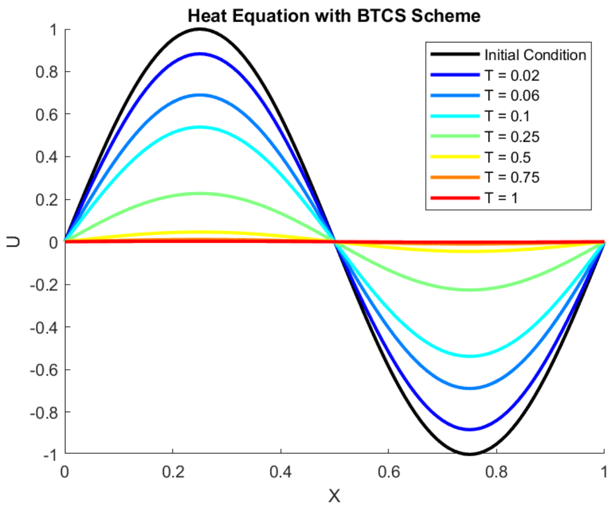
Performing Von Neumann stability analysis on (1):

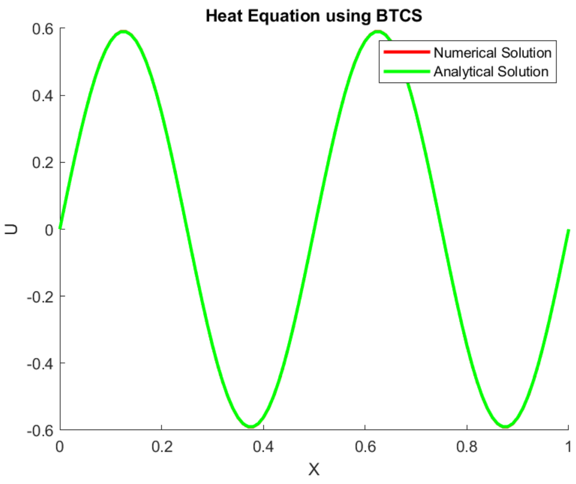
$$\begin{bmatrix} 1+2\lambda & -\lambda & 0 & 0 & \dots & 0 & 0 \\ -\lambda & 1+2\lambda & -\lambda & 0 & \dots & 0 & 0 \\ 0 & -\lambda & 1+2\lambda & -\lambda & \dots & 0 & 0 \\ 0 & 0 & -\lambda & 1+2\lambda & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1+2\lambda & -\lambda \\ 0 & 0 & 0 & 0 & \dots & -\lambda & 1+2\lambda \end{bmatrix} \begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ u_4^{n+1} \\ \vdots \\ u_{m-1}^{n+1} \\ u_m^{n+1} \end{bmatrix} = \begin{bmatrix} u_1^n \\ u_2^n \\ u_3^n \\ u_4^n \\ \vdots \\ u_{m-1}^n \\ u_m^n \end{bmatrix}.$$

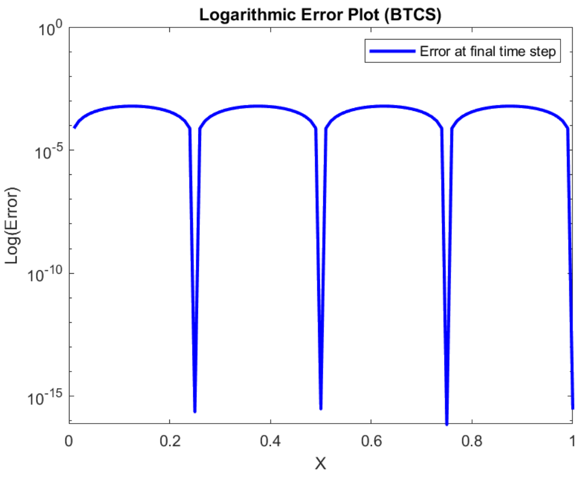
$$\xi = \frac{1}{-\lambda(e^{-ik\Delta x}+e^{ik\Delta x})+(1+2\lambda)}$$

$$= \frac{1}{1-2\lambda(1-cos\ cos\ (k\Delta x))} \text{———}(2)$$

On analyzing every edge case on (2) we can conclude that for every case $|\xi| \leq 1$. Therefore BTCS is unconditionally stable when applied to the hear equation.







Conclusion

It was shown in the study that numerical schemes particularly the Lax Wendroff explicit method can be effectively employed in approximating the solutions of the heat equation. When small time steps and high spatial accuracy were employed it was found out that numerical results were in good agreement with the analytical solution but as time steps become bigger the discrepancies grew. Logarithmic error analysis demonstrated that it is indeed the case that by increasing the resolution in the number of grid points, the errors are unwound and the scheme is consistent with convergence.

It was the stability however that was of prime concern as the Lax-Wendroff scheme was stable for all time steps meeting the Courant-Friedrichs-Lewy (CFL) condition. In addition, the Lax Equivalence Theorem has also shown that the two together consistency and stability ensure convergence and thence reliable outcomes when the scheme is used properly. This balance between the abstract properties and the hands on outcomes is further justified as useful.

As a conclusion, schemes such as Lax-Wendroff explicit methods can be used as relief when solving various types of partial differential equations such as the heat conjugation in this case. The work also considers the matter of how essential it is to definite the correctness of the selected scheme since optimal results can only be achieved through balance of computational load, stability and accuracy.

REFERENCES

- Numerical Partial Differential Equations: Finite Difference Methods, by JW Thomas
- Elements of Partial Differential Equations, International Series in Pure and Applied Mathematics, by Ian N. Sneddon
- Numerical Methods in Engineering and Science, B.S Grewal
- YouTube: Steve Brunton - https://www.youtube.com/watch?v=Jt5R-Tm8cV8&list=PLMrJAkhIeNNQromC4WswpU1krLOq5Ro6S
- YouTube: Aerodynamic CFD - https://www.youtube.com/playlist?list=PLcqHTXprNMINSc1n62_-SYUF963y_vYTT
- YouTube: NPTEL - https://www.youtube.com/playlist?list=PLq-Gm0yRYwTizWtb_xwk0KEMzcoeLbOZq

MATLAB CODE: Drive folder