

ADDIS ABABA UNIVERSITY
COLLEGE OF NATURAL SCIENCE
DEPARTMENT OF MATHEMATICS



Graduate project report on
Explicit Finite Difference Schemes using Ghost points
for the Heat equation with Insulated ends.
Submitted in partial fulfilment of requirements for the
Degree of Master of Science in Mathematics

Prepared by: Tesfu Tezera
Advisor: Tadesse Abdi(Ph.D)

June, 2015
Addis Ababa, Ethiopia

Contents

Acknowledgement	i
Abstract	ii
Notation	iii
Introduction	iv
1 Preliminaries	1
1.1 The Heat equation of an Insulated Bar	1
1.2 Finite Difference Method	1
1.2.1 Analysis of the Finite-Difference Model	2
1.2.2 Outline of the Method	4
1.3 Discretization of the Domain	4
1.4 Ghost point and associated Ghost values	6
1.5 Boundary Conditions	6
1.5.1 Neumann Boundary Conditions	7
1.6 Driving Finite Difference Approximation	8
1.6.1 Taylor's Theorems Applied to the Finite Difference Methods	8
2 Explicit Finite Difference Scheme	10
2.1 Discretizing the Heat equation of an Insulated Bar	11
2.2 Boundary conditions of Heat equation of a Bar with an Insu- lated Ends	14
2.3 Discretization of the Boundary Conditions	14
2.4 Discretization of the Initial Conditions	15
3 Consistency, Stability and Convergence	17
3.1 Consistency	17
3.2 Stability	19
3.2.1 CFL condition	19

3.2.2	Von Neumann stability analysis	20
3.3	Convergence	22
4	Implementation	24
4.1	Test Problem	24
4.2	Implementation	25
	Summary	29
	Bibliography	30

Addis Ababa University
Department of Mathematics

The undersigned hereby certify that they have read and recommend to the school of graduate studies for acceptance of a project entitled **Explicit finite Difference Schemes using Ghost points for the heat equation with insulated ends**, by Tesfu Tezera in partial fulfillment of the requirements for the degree of master of Science.

Dated: June, 2015

Advisor:

Dr.Tadesse Abdi

Examiner 1.Dr. _____

Examiner 2.Dr. _____

June, 2015
ADDIS ABABA ,ETHIOPIA

Acknowledgement

I would like to express my heart-felt gratitude to my advisor Dr.Tadesse Abdi who devoted much of his time to read through the drafts of my project paper and made constructive suggestion and comments.

I want to extend sincere and special thanks to my brother Ato Meshesha Kibret for his unreserved material and moral support throughout my work. I am also deeply indebted to my families and parents for inspiring and encouraging me in my plans.

Abstract

In this project report, explicit finite difference scheme for the linear heat equation in an insulated bar in one space dimension is considered. This method is used to solve the partial derivatives in the partial differential equations at each grid point that are derived from neighboring values by using Taylors theorem . The forward - time centered - space(FTCS) and explicit schemes are developed. To find the boundary equations of the schemes ghost points are introduced in central difference approximation . The MATLAB implementation allow the readers to experiment with the stability limit of the forward time , centered space(FTCS)schemes

key words- finite difference, explicit scheme, heat equation in an insulated bar , Ghost point

Notation

$\mathbb{R}$	The set of all real numbers.
FDM	Finite difference method .
$1d$	One dimension.
$U = U(x, t)$	The temperature at point x at time t .
c	The rate of diffusion.
h_t	The local distance between adjacent time steps.
h_x	The local distance between adjacent points in space.
$O(h^n)$	Discretization error or Truncation error.

Introduction

The finite difference approximations for derivatives are one of the simplest and of the oldest methods to solve differential equations. It was already known by L. Euler (1707-1783) ca. 1768, in one dimension of space and was probably extended to dimension two by C. Runge (1856-1927) ca. 1908. The advent of finite difference techniques in numerical applications began in the early 1950s and their development was stimulated by the emergence of computers that offered a convenient framework for dealing with complex problems of science and technology. Theoretical results have been obtained during the last five decades regarding the accuracy, stability and convergence of the finite difference method for partial differential equations.

In this paper, We will introduce the explicit finite difference method to solve the heat equation using ghost point in an insulated bar.

The basic idea in the finite difference methods (FDM) is approximation of derivatives by difference quotients thereby replacing the derivatives involved in a given differential equation (PDE or ODE) by difference quotients at the grid points and consequent reduction of a given differential equation into a system of algebraic equations.

We recall that a PDE involves several independent variables (at least 2) and also requires some sort of regularity of the solution , u on domain Ω . In particular $C^n(\Omega)$ smoothness of a solution u of an n^{th} order PDE is required to enable Taylor series expansion of u at each interior point of the domain , Ω^0 .

Discretization of the domain, Ω takes into account all the independent variables in a PDE . We need to choose a step size corresponding to each independent variable. For the sake of demonstration , if we consider the domain , Ω to be a unit square,

$$\Omega = (0, 1) \times (0, 1)$$

we can generate rectangular grids

$$\Omega_h = \{(x_i, t_j) : x_i = (i-1)h_x, \quad t_j = (j-1)h_t\},$$

$$i = 1, \dots, n \quad j = 1, \dots, m$$

$$\text{with } h_x = \frac{1}{n-1} \text{ and } h_t = \frac{1}{m-1}$$

we denote the unknown function u at the grid point (x_i, t_j) by u_i^j , i.e $u_i^j = u(x_i, t_j)$

Consequently the finite difference formulation of 1d heat equation with Cauchy data

$$u_t - u_{xx} = 0, 0 < t, x < 1$$

$$u(x, 0) = f(x); 0 \leq x \leq 1$$

begins with difference approximation of u_t and u_{xx} at the grid points as follows,

$$u_t(x_i, t_j) = \frac{u_i^{j+1} - u_i^j}{h_t}$$

$$u_{xx}(x_i, t_j) = \frac{u_{i+1}^j - 2u_i^j + u_{i-1}^j}{h_x^2}$$

This leads to the difference equation

$$\frac{u_i^{j+1} - u_i^j}{h_t} - \frac{u_{i+1}^j - 2u_i^j + u_{i-1}^j}{h_x^2} = 0$$

further simplification yields

$$u_i^{j+1} - \lambda u_{i+1}^j + (2\lambda - 1)u_i^j - \lambda u_{i-1}^j = 0$$

$$\text{where } \lambda = \frac{h_t}{h_x^2}$$

This is a 4-point stencil that can be denoted by stencil symbol

$$\begin{pmatrix} & 1 & \\ -\lambda & 2\lambda - 1 & -\lambda \end{pmatrix}$$

Evidently, the stencil symbol dictates that this finite difference approximation is explicit scheme that allows marching forward (advancing) in time. We begin this chapter by considering the model of heat equation with an insulated ends and then setting the outline how to discretize using Taylor's Theorem to derive finite difference approximations to the first and second derivatives of a function.

After the basic concepts have been introduced, we begin our discussion of finite difference schemes. The important concepts of convergence, consistency, and stability are presented and shown to be related by the Lax equivalence theorem.

Chapter 1

Preliminaries

1.1 The Heat equation of an Insulated Bar

An insulated bar means there's no heat flow through the ends. In derived heat equation from physical considerations the change of heat flow is U_x . So, change of heat flow is zero at the two ends and this will happen when

$$U_x(0, t) = 0 = U_x(L, t) \quad , for \quad t > 0$$

The one dimensional heat equation of an insulated bar is

$$U_t = C^2 U_{xx}, \quad 0 < x < L, \quad t > 0 \quad (1.1)$$

$$U_x(0, t) = 0 \quad and \quad U_x(L, t) = 0, \quad t > 0 \quad (1.2)$$

$$U(x, 0) = f(x), 0 \leq x \leq L \quad (1.3)$$

Where $U = U(x, t)$ is the temperature at point x at time t and C is the change of diffusion and depends on the physical properties such as specific heat, conductivity, density etc . Equation (1.1) is Model of heat equation in a slab of material with length L . The solution U to this partial differential equation gives the temperature distribution of a function of both space and time.. Our goal is to compute the solution of equation (1.1) that requires specification of boundary conditions at $x = 0$ and $x = L$.

1.2 Finite Difference Method

Because of the importance of the diffusion or heat equation to a wide variety of fields, there are many analytical solutions of that equation for a wide variety of initial and boundary conditions. However, one very often runs into

a problem whose particular conditions have no analytical solution, or where the analytical solution is even more difficult to implement than a suitably accurate numerical solution. Here we will discuss one particular method for numerical solution of partial differential equations called the finite difference method.

A finite difference method proceeds by replacing the derivatives in the differential equations with finite difference approximations. This gives a large but finite algebraic system of equations to be solved in place of the differential equation, something that can be done on a computer.

It is simple to code and economic to compute. In a sense, a finite difference formulation offers a more direct approach to the numerical solution of partial differential equations than does a method based on other formulations. It consists in approximating the differential operator by replacing the derivatives in the equation using difference quotients. The drawback of the finite difference methods is accuracy and flexibility.

The finite difference approach is one of the premier mathematical tools employed to solve partial differential equations. It is a means of obtaining numerical solutions to partial differential equations. The most common finite difference methods for the solution of partial differential equation can be explicit or implicit .

These are closely related but differ in stability, accuracy and execution speed. In this project report, I have illustrated how one can solve the solution of 1d heat equation in an insulated bar using an explicit finite difference scheme.

1.2.1 Analysis of the Finite-Difference Model

In the limit as the mesh spacing (h_x and h_t) go to zero, the numerical solution obtained with any useful scheme will approach the true solution to the original differential equation. However, the rate at which the numerical solution approaches the true solution varies with the scheme. The error between the numerical solution and the exact solution is determined by the error that is committed by going from a differential operator to a difference operator. This error is called local discretization error or truncation error. The term truncation error reflects the fact that a finite part of a Taylor series is used in the approximation.

Every finite difference approximation of a partial differential equation should

possess consistency, convergence and stability.

- Consistency: A finite difference schemes for partial differential equation is consistent , if the difference between partial differential equation and finite difference equation vanishes as the interval(h_x) and the time step size (h_t)approaches zero. Consistency deals with how well the finite difference equation approximates the partial differential equation.
- By stability we mean that errors at one stage of the calculations do not cause increasingly large errors as the computations are continued.
- Convergence : It means that the solution to the finite difference equation approaches the true solution to the partial differential equation as both the grid interval and time step size are reduced (as h_x and h_t tends to zero).The necessary and sufficient conditions for convergence are consistency and stability.

These three factors that characterize a numerical scheme are linked together by Lax equivalence theorem (Frankel and Du Fort, 1953) which states that given a well-posed linear initial value problem and a consistent finite difference scheme, stability is the necessary and sufficient condition for convergence.

In general, a problem is said to be well-posed if:

- A solution to the problem exists.
- The solution is unique when it exists.
- The solution depends continuously on the problem data.

Grid (Mesh)

The mesh (grid) is the set of locations where the discrete solution is computed. These points are called nodes, and if one were to draw lines between adjacent nodes in the domain the resulting image would resemble a net or mesh. Two key parameters of the mesh are h_x the local distance between adjacent points in space, and h_t the local distance between adjacent time steps. Consider a computational domain in the two-dimensional space of variables (x, t) . Cover this space by a grid of discrete points(x_i, t_j) given by $x_i = x_o + ih_x$, $t_j = t_o + jh_t$, $i = 0, 1, 2, \dots, j = 0, 1, 2, \dots$. Here, h_x is usually called grid spacing, and h_t is called time step since t usually represents time. A spatial grid that is the most appropriate for the problem under consideration should be chosen. In many applications, the regular (uniform) grid with the grid spacing h_x is a natural and reasonable choice

1.2.2 Outline of the Method

When solving the one-dimensional heat equation, it is important to understand that the solution $u(x, t)$ is a function of two variables. The domain of $u(x, t)$ is a subset of the xt -plane. More specifically, the domain is $\{(x, t) : 0 \leq x \leq L \text{ and } 0 \leq t < \infty\}$ or $\{(x, t) : (x, t) \in [0 : L] \times [0, \infty)\}$. When we find an exact solution also known as a closed-form solution, we find $u(x, t)$ for every pair (x, t) in $[0, L] \times [0, \infty)$. However, when we solve partial differential equation numerically, the continuous xt -plane (infinitely many points) is replaced by a finite set of grid points, then a partial differential equation is replaced by algebraic equations and the algebraic equations are solved at each grid point. More specially, follow the following steps:

1. Discretize the domain. During this operation, the continuous xt -plane is replaced by a discrete domain, a finite set of points(grids)
2. Discretize the partial differential equation. During this operation, all the partial derivatives which appear in the partial differential equation are approximated by finite differences. During this process, the partial differential equation is replaced by algebraic equations.
3. Discretize the boundary conditions and the initial conditions. This step is similar to step 2.
4. By solving the algebraic equations obtained in steps 2 and 3 at the points we obtain the solution at grid points.

1.3 Discretization of the Domain

The first step is to discretize the domain (xt -plane). By dividing the interval $[0, L]$ of the x -axis into $n+1$ points equally spaced. By labeling these points $x_0, x_1, x_2, \dots, x_n$. Let $h_x = \Delta x = \frac{L}{N}$ denote the distance between two consecutive points. Then we have $x_i = ih_x$. Similarly, By dividing the interval $(0, L)$ of the t -axis into points equally spaced. Label these points $t_0, t_1, t_2 \dots$. Let $h_t = \Delta t$ be the distance between two consecutive points. Then we have $(t_j = jh_t)$. We will solve the partial differential equation at the points (x_i, t_j) for $i = 0, 1, 2, \dots, n$, $j = 0, 1, 2, \dots$. we also use the following notation:

$$\begin{aligned} U(x_i, t_j) &= U_i^j \\ U(x_{i+1}, t_j) &= U_{i+1}^j \\ U(x_{i-1}, t_j) &= U_{i-1}^j \end{aligned}$$

$$\begin{aligned} U(x_i, t_{j+1}) &= u_i^{j+1} \\ U(x_i, t_{j-1}) &= U_i^{j-1} \end{aligned}$$

The computational domain in fig.1 contains an infinite number of x values so first we must replace them by finite set .This is called spatial discrization .



Fig.1. One dimensional computational domain

For simplicity the computational domain is replaced by a grid of N equally spaced grid points. The first grid point at $x = a$ and ending with the last grid point at $x = b$ the constant grid spacing h_x is

$$h_x = \Delta x = \frac{b-a}{N}$$

The values of x in the discretised computational domain are indexed by subscripts to give

$$x_0 = a, x_1 = a + h_x, \dots, x_i = a + ih_x, \dots, x_N = a + Nh_x = b$$

Since the grid spacing is constant

$$x_{i+1} = x_i + h_x$$

the discretised computational domain is shown in fig.2.below



Fig.2. Discretised computational domain

1.4 Ghost point and associated Ghost values

When solving a partial differential equation using a finite difference scheme we may need to specify ghost grid points and associated ghost values for the dependent variable . In explicit finite difference scheme to solve one dimensional heat equation we may need ghost point to the left of the first grid point and to the right of the last grid point as illustrated in the following diagram.

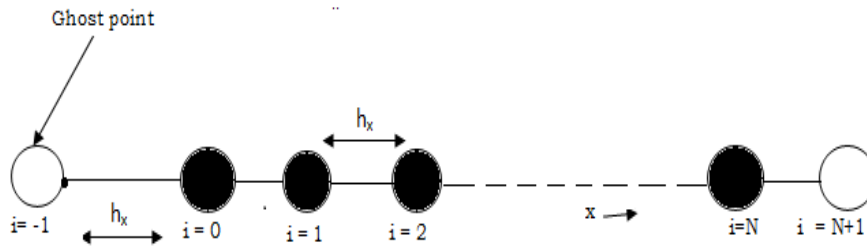


Fig3. Ghost (white) and grid (black) points for the FDM schemes

Note:

- In general in one dimensional region on which the grid points are indexed by $i = 0, 1, 2, 3, \dots, N$. we will index a left ghost point $i = -1$ and a right ghost point by $i = N + 1$
- Grid points indexed by $i = 0, 1, 2, 3, \dots, N$ in the x direction . Indices -1 , and $N + 1$ indicates ghost point

1.5 Boundary Conditions

Heat equation in a bounded domain is prescribed along with a boundary condition. Commonly used boundary condition are ,

- Dirichlet (displacement)
- Neumann (flux) and
- Robin (mixed) boundary conditions

The Dirichlet boundary condition is relatively easy in that the solution , u is given along the boundary $\partial\Omega$ of the domain Ω

$$\text{i.e } u(x, t)|_{\partial\Omega} = u_o(x, t)$$

For the Neumann boundary condition , the normal derivatives

$$\frac{\partial u}{\partial n} = \nabla u \cdot n$$

is given along the boundary ,where n is the unit outward normal vector. Finally ,Robin boundary condition takes the form

$$(\partial u + \beta \frac{\partial u}{\partial n})|_{\partial\Omega} = g$$

In solving PDE with Neumann or Robin data we need to introduce fictitious grid points next to the grid points on boundary ,just to deal with the boundary data that involve derivatives . This fictitious discretization points are what we call ghost points.

1.5.1 Neumann Boundary Conditions

Neumann Boundary Conditions appear when the boundary is insulated

$$\frac{\partial u}{\partial n} = 0$$

or when heat is transferred by radiation into surrounding medium (whose temperature is v)

$$-K \frac{\partial u}{\partial n} = H(u - v)$$

Where H is the coefficient of surface heat transfer and K the internal conductivity of the material.

Here we show how to approximate those two types of boundary conditions in connection with the one dimensional heat equation.

$$u_t = K u_{xx} \quad 0 < x < 1$$

$$u(0, t) = g(t)$$

$$\frac{\partial u(L, t)}{\partial n} = -H(u(L, t) - v)$$

$$u(x, 0) = f(x)$$

Clearly one can use backward difference to approximate the Neumann boundary condition on the left end ($x=1$). But this is first order which will degrade the accuracy in x every where (since the error will propagate to the interior in time) . If we decide to use first order central approximation ,then we have

$$\frac{u_{N+1}^n - u_{N-1}^n}{2h_x} = -h(u_N^n - v) \quad (1.4)$$

Where x_{N+1} is fictitious point outside the interval i.e $x_{N+1} = 1 + h_x$. This will require another equation to match the number of unknowns. we then apply the finite difference equation at the boundary. For example if we are using explicit scheme then we apply the equation

$$u_j^{n+1} = \lambda u_{j-1}^n + (1 - 2\lambda)u_j^n + \lambda u_{j+1}^n, \quad (1.5)$$

for $j = 1, 2, \dots, N$, At $j = N$,we then have

$$u_N^{n+1} = \lambda u_{N-1}^n + (1 - 2\lambda)u_N^n + \lambda u_{N+1}^n \quad (1.6)$$

substitute the values of u_{N+1}^n from 1.4 in to 1.6 and we get

$$u_N^{n+1} = \lambda u_{N-1}^n + (1 - 2\lambda)u_N^n + \lambda[u_{N-1}^n - 2h_x(u_N^n - v)], \quad (1.7)$$

1.6 Driving Finite Difference Approximation

The finite difference methods work by replacing the region over which the independent variables in the partial differential equations are defined by finite grid (also called mesh) of points at which the independent variable is approximated. The partial derivatives in the partial differential equations at each grid point are approximated from neighboring values by using Taylors theorem

1.6.1 Taylor's Theorems Applied to the Finite Difference Methods

In this section we discuss finite difference approximation to partial derivatives . The approximations are based on Taylor series expansions of a function of one or more variables .

Recall that the Taylor series expansion for a function of one variable is given by

$$U(x + h, y) = U(x, y) + \frac{h}{1!}U_x(x, y) + \frac{h^2}{2!}U_{xx}(x, y) + \dots$$

the remainder is given by

$$o(h)^n = U^{(n)}(\xi) \frac{h^n}{n!}, \quad \xi \in (x, x+h)$$

For a function of more than one independent variable . We have the derivatives replaced by partial derivatives . We give here in the case of 2 independent variables

$$U(x+h, y+k) = U(x, y) + \frac{h}{1!}U_x(x, y) + \frac{k}{1!}U_y(x, y) + \frac{h^2}{2!}U_{xx}(x, y) + \frac{2hk}{2!}U_{xy}(x, y) + \frac{h^2}{2!}U_{yy}(x, y) + \frac{h^3}{3!}U_{xxx}(x, y) + \frac{3h^2k}{3!}U_{xxy}(x, y) + \frac{3hk^2}{3!}U_{xyy}(x, y) + \frac{k^3}{3!}U_{yyy}(x, y) + \dots$$

the remainder can be written in the form

$$\frac{1}{n!}(h \frac{\partial}{\partial x})^n U(x + \theta h, \theta k). \quad 0 \leq \theta \leq 1$$

Here we used a subscript to denote partial differentiation . we will be interested to obtain approximation about the point (x_i, y_j) and we used a subscript to denote the function values at point $U_i^j = U(x_i, y_j)$

The Taylor series expansion for U_{i+1} about the point x_i is given by

$$U_{i+1} = U_i + h_x U_x(x_i) + \frac{h_x^2}{2!} U_{xx}(x_i) + \frac{h_x^3}{3!} U_{xxx}(x_i) + \dots$$

The Taylor series expansion for U_{i+1}^{j+1} about the point (x_i, y_j) is given by

$$U_{i+1}^{j+1} = U_i^j + (h_x U_x + h_y U_y)_i^j + (\frac{h_x^2}{2} U_{xx} + h_x h_y U_{xy} + \frac{h_y^2}{2} U_{yy})_i^j + \dots$$

Several difference representations can be found for the partial derivatives of $U(x, y)$. Let us use the following operator

First order forward difference $U_x(x_i, y_j) = \frac{U_{i+1}^j - U_i^j}{h_x}$

First order backward difference $U_x(x_i, y_j) = \frac{U_i^j - U_{i-1}^j}{h_x}$

First order centered difference $U_x(x_i, y_j) = \frac{U_{i+1}^j - U_{i-1}^j}{2h_x}$

Second order centered difference $U_{xx}(x_i, y_j) = \frac{U_{i+1}^j - 2U_i^j + U_{i-1}^j}{h_x^2}$

In a similar fashion we can define the corresponding operator in y.

Chapter 2

Explicit Finite Difference Scheme

Consider the 1d transient heat equation without source term,

$$\rho C_p \frac{\partial u}{\partial t} - \frac{\partial}{\partial x} (k \frac{\partial u}{\partial x}) = 0$$

Evidently , if the density (ρ), the heat capacity (C_p) and the thermal conductivity (k) all are constant over the domain Ω , the above equation reduce to

$$u_t - \alpha^2 u_{xx} = 0$$

with $\alpha^2 = \frac{k}{\rho c_p}$ representing the thermal diffusivity. The forward - time centered - space scheme for this problem is

$$\frac{u_i^{j+1} - u_i^j}{h_t} - k \frac{u_{i+1}^j - 2u_i^j + u_{i-1}^j}{h_x^2} = 0$$

which upon rearrangement ,reduce to

$$u_i^{j+1} = u_i^j + k h_t \frac{u_{i+1}^j - 2u_i^j + u_{i-1}^j}{h_x^2}$$

This is a time marching scheme that enables one to compute the solution at the current step (j+1) from the known values to the proceeding step (j) the pertinent stencil symbol of this scheme is

$$\begin{pmatrix} & 1 & \\ -\lambda & 2\lambda - 1 & -\lambda \end{pmatrix}$$

where $\lambda = k \frac{h_t}{h_x^2}$
such a scheme is called **explicit** and the corresponding stability condition is given by $0 < \lambda < \frac{1}{2}$.

The backward - time centered space scheme for the same problem is

$$\frac{u_i^{j+1} - u_i^j}{h_t} - k \frac{u_{i+1}^{j+1} - 2u_i^{j+1} + u_{i-1}^{j+1}}{h_x^2} = 0$$

once again, setting $\lambda = k \frac{h_t}{h_x^2}$ rearranging of this digitized equation leads to

$$-\lambda u_{i-1}^{j+1} + (1 + 2\lambda)u_i^{j+1} - \lambda u_{i+1}^{j+1} = u_i^j$$

the corresponding stencil symbol of this is

$$\begin{pmatrix} -\lambda & 2\lambda + 1 & -\lambda \\ & -1 & \end{pmatrix}$$

In this case the solution for the current step $(j+1)$ at a grid point (x_i, t_{j+1}) can not be computed from the values of the preceding step(j) alone.

The contribution of the neighbouring grids at current step $(j+1)$, namely at (x_{i-1}, t_{j+1}) and (x_{i+1}, t_{j+1}) is needed . As a result of this, we need to solve linear system of equations, such a scheme is what we call **implicit** . For implicit scheme there is no stability problem.

Let us recall the new heat boundary value problem for an insulated bar with end held at a temperature 0.

$$U_t = c^2 U_{xx}, \quad 0 < x < L, t > 0$$

$$U_x(0, t) = 0, \quad U_x(L, t) = 0, \quad t > 0$$

$$U(x, 0) = f(x), \quad 0 \leq x \leq L$$

2.1 Discretizing the Heat equation of an Insulated Bar

The idea is to discretize the problem by choosing a step size h_x in x and step size h_t in t and trying to approximate the temperature U on grid of the points in the xt -plane. Say we want to approximate the values of U at $n-1$ equally spaced points between 0 and L

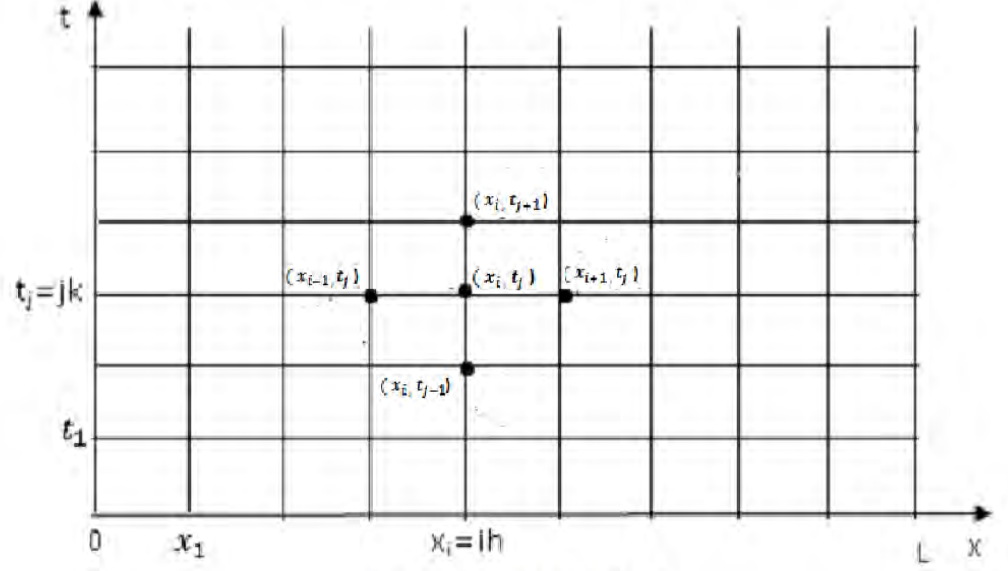


Fig 4. Discretized Grid

and divide the interval from 0 to L in to n parts by choosing $x = h_x, 2h_x, 3h_x, \dots, (n-1)h_x$. Similarly, we pick a h_t and consider only the time values $t = 0, h_t, 2h_t, \dots$. Thus, we want to develop a numerical scheme that yields approximation for the values. Discretize time such that at the n^{th} time step, the variable time is approximated as $t_j = jh_t$, $j = 1, 2, 3, n-1$

where h_t is the time interval. Thus the temperature evaluated at a given grid point at a given time step is given by $U(x_i, t_j)$

$$U_i^j = U(ih, jk), \text{ where } 0 < i < N \text{ and } j \geq 0 \quad (2.1)$$

Then a Taylor's Series expansion of the function $U(x_i + h_x, t_j)$ about the grid coordinate (x_i, t_j) is

$$U(x_i + h_x, t_j) = U(x_i, t_j) + h_x U_x(x_i, t_j) + \frac{h_x^2}{2!} U_{xx}(x_i, t_j) \dots, \quad (2.2)$$

Similarly, the Taylor's Series expansion of the function $U(x_i, h_t + t_j)$ about the grid location (x_i, t_j) is

$$U(x_i, h_t + t_j) = U(x_i, t_j) + h_t U_t(x_i, t_j) + \frac{(h_t)^2}{2!} U_{tt}(x_i, t_j) \dots, \quad (2.3)$$

In an analogous manner we can construct forward difference for the time derivative and central difference approximation for the spatial derivatives.

$$U_{xx}(x_i, t_j) = \frac{U_{(i+1)}^j - 2U_i^j + U_{(i-1)}^j}{(h_x)^2} \quad (2.4)$$

Where the error in the approximation is proportional to $(h_x)^2$. Similarly, the one-sided finite difference approximation (2.5) is used to approximate the time derivative, and so

$$U_t(x_i, t_j) = \frac{U_i^{j+1} - U_i^j}{h_t} \quad (2.5)$$

Where the error is proportional to h_t . In practice, one should try to ensure that the approximations have similar orders of accuracy, which leads us to choose

$$h_t \approx (h_x)^2$$

Assuming $h_x < 1$, this requirement has the important consequence that the time steps must be much smaller than the space mesh size.

The finite difference approximation of Heat equation (1.1) becomes

$$\frac{U_i^{j+1} - U_i^j}{h_t} - C^2 \frac{U_{(i+1)}^j - 2U_i^j + U_{(i-1)}^j}{(h_x)^2} = 0 \quad (2.6)$$

Rearranging equation 2.6 and solving for this U_i^{j+1} gives:

$$U_i^{j+1} = U_i^j + C^2 \frac{h_t}{(h_x)^2} (U_{(i+1)}^j - 2U_i^j + U_{(i-1)}^j) \quad (2.7)$$

Equation (2.7) is called the Forward Time, Centered Space approximation to the heat equation. A slight improvement in computational efficiency can be obtained with a small rearrangement of Equation (2.7)

$$U_i^{j+1} = (1 - 2\lambda)U_i^j + \lambda(U_{(i+1)}^j + U_{(i-1)}^j) \quad (2.8)$$

$$i = 1, 2, 3, \dots, n-1, \quad j = 0, 1, 2, \dots$$

$$\text{Where } \lambda = C^2 \frac{h_t}{(h_x)^2}$$

Equation 2.8 allows us to calculate all the values of U at a step $j+1$ using the values at a step j .

2.2 Boundary conditions of Heat equation of a Bar with an Insulated Ends

For the Neumann boundary conditions,

$$U_x(0, t) = 0 = U_x(L, t) \quad \text{for } t > 0$$

The values of the solution on the boundary points of the grid are not immediately available, so one needs to use a difference approximation for the conditions in order to march forward in time. Using the forward or backward differences to approximate U_x on the boundary would introduce a local truncation error of order $O(h_x)$, which is bigger than the truncation error $O(h_x)^2$, thus contaminating the numerical solution. To have errors of the same order as the difference equation, we must use the centered differences. For these we introduce "ghost points" $U_{(-1)}^j$ and $U_{(i+1)}^j$ in addition to the grid points $U_o^j, U_1^j, \dots, U_n^j$.

From the above identities we can compute the values $U_{(-1)}^j$ and $U_{(i+1)}^j$ at the ghost points at time level j , which will then be used to compute U_0^{j+1} and U_n^{j+1} thus arriving at numerical values at the boundaries.

2.3 Discretization of the Boundary Conditions

The point $(0, t)$ is replaced on the discrete domain by $(0, t_j) = (x_0, t_j)$ and the value of the function U at that point is denoted U_0^j . Similarly, (L, t) is replaced on the discrete domain by (x_i, t_j) and the value of the function U at this point is denoted U_i^j . Hence, the boundary conditions become $U_0^j = 0$ and $U_n^j = 0$ for $j = 1, 2, \dots$. In particular this means U_i^j is unknown only for $i = 1, 2, 3, \dots, n-1$. We can derive it from what we are given by discretizing U_x using the centered difference approximation as follows,

From first order centered difference approximation

$$U_x(x_i, t_j) \approx \frac{U_{(i+1)}^j - U_{(i-1)}^j}{2h_x}$$

At the point $(0, t)$ for any $t, i = 0$, Thus

$$U_x(0, t) \approx \frac{U_{(i+1)}^j - U_{(i-1)}^j}{2h_x}$$

The condition $U_x(0, t) = 0$ gives:

$$\frac{U_{(i+1)}^j - U_{(i-1)}^j}{2h_x} = 0$$

$$\begin{aligned}
U_{(i+1)}^j - U_{(i-1)}^j &= 0, \quad \text{for } i = 0 \\
U_1^j &= U_{(-1)}^j
\end{aligned} \tag{2.9}$$

substituting Equation 2.9 into Equation 2.8 gives:

$$U_0^{j+1} = (1 - 2\lambda)U_0^j + \lambda(U_1^j + U_{(-1)}^j)$$

and since $U_1^j = U_{(-1)}^j$,

$$\begin{aligned}
U_0^{j+1} &= (1 - 2\lambda)U_0^j + \lambda(U_1^j + U_{(-1)}^j) \text{ becomes} \\
U_0^{j+1} &= (1 - 2\lambda)U_0^j + 2\lambda U_1^j, \quad \text{for } j \geq 0
\end{aligned} \tag{2.10}$$

From boundary condition $U_x(L, t) = 0$, we have

$$\frac{U_{(n+1)}^j - U_{(n-1)}^j}{2h_x} = U_x(L, t) = 0$$

This gives:

$$\begin{aligned}
U_{(n+1)}^j - U_{(n-1)}^j &= 0 \\
U_{(n+1)}^j &= U_{(n-1)}^j
\end{aligned} \tag{2.11}$$

Substituting Equation 2.11 into Equation 2.8, becomes

$$U_n^{j+1} = (1 - 2\lambda)U_n^j + \lambda(U_{(n+1)}^j + U_{(n-1)}^j)$$

and using Equation 2.11 the above Equation becomes

$$U_n^{j+1} = (1 - 2\lambda)U_n^j + 2\lambda U_{(n-1)}^j \tag{2.12}$$

2.4 Discetization of the Initial Conditions

The point $(x, 0)$ is replaced on the discrete domain by $(x_i, 0) = (x_i, t_0)$ and the value of the function at that point is denoted $U_{(i,0)}$. Hence, the initial conditions becomes

$$U_i^0 = U(ih_x, 0) = f(ih_x) \quad \text{for } i = 0, 1, 2, \dots, n \tag{2.13}$$

Finally we need these boundary values as we use our difference equations (2.8), to step forward in time. Since to compute U_i^{j+1} . We typically need this three values $U_{(i-1)}^j$, U_i^j and $U_{(i+1)}^j$ thus when $i = 1$ and $n - 1$ we use U_0^{j+1} and U_n^{j+1} respectively

The values of U on the x-axis are determined by the initial conditions and the values of U on the vertical line $x = 0$ (*the t - axis*) and $x = L$ are determined by the boundary conditions. All other values of U on the grid are computed from Equation 2.8.

we write equation 2.8 in the matrix form

$$\begin{pmatrix} U_1^{j+1} \\ U_2^{j+1} \\ \vdots \\ U_{n-1}^{j+1} \end{pmatrix} = \begin{pmatrix} 1-2\lambda & \lambda & & & \\ \lambda & 1-2\lambda & \lambda & & \\ & \lambda & 1-2\lambda & \lambda & \\ & & \ddots & \ddots & \ddots \\ & & & \lambda & 1-2\lambda \end{pmatrix} \begin{pmatrix} U_1^j \\ U_2^j \\ \vdots \\ U_{n-1}^j \end{pmatrix}$$

or

$$U^{j+1} = AU^j \tag{2.14}$$

where λ is defined by equation 2.8

$$U^j = [U_1^j, U_2^j, \dots, U_{M-1}^j]^T$$

and A is the tridiagonal matrix.

Chapter 3

Consistency, Stability and Convergence

3.1 Consistency

A discrete approximation to a partial differential equation is said to be consistent if in the limit of the step size(s) going to zero, the original partial differential equation system is recovered, i.e. the truncation error approaches zero.

The local truncating error is the difference between the solution of the partial differential equation and its finite difference approximation. If the local truncation error tends to zero as the step size goes to zero. In this case the method is said to be consistent.

Definition 3.1.1. Consider a differential equation $Pu = 0$ and a finite difference approximation of it $P_\Delta U_i^j = 0$. Let $v(x, t)$ be any smooth function, then the local discretization or local truncation error is.

$$\tau_i^j = Pv(ih_x, jh_t) - P_\Delta v(ih_x, jh_t) \quad (3.1)$$

Remark 3.1.1. P is a differential operator, $Pv = v_t - c^2 v_{xx}$ is for the heat equation (1.1). Similarly, P_Δ is a finite difference operator; thus, the finite difference operator for the heat operator is

$$P_\Delta v = \frac{v_i^{j+1} - v_i^j}{h_t} - c^2 \frac{v_{i+1}^j - 2v_i^j + v_{i-1}^j}{(h_x)^2}$$

Remark 3.1.2. The function v is often regarded as the exact solution u of the partial differential equation. This is convenient but not necessary; thus, v may be any smooth function. This gives us a way of defining the local discretization error even when the solution of the partial differential equation has singularities.

The local discretization error of the finite difference approximation for the heat equation is

$$\begin{aligned}\tau_i^j &= (v_t - c^2 v_{xx})_i^j - \left(\frac{v_i^{j+1} - v_i^j}{h_t} - c^2 \frac{v_{i+1}^j - 2v_i^j + v_{i-1}^j}{(h_x)^2} \right) \\ &\quad \text{or} \\ \tau_i^j &= (v_t)_i^j - \frac{v_i^{j+1} - v_i^j}{(h_t)} - c^2 \left[(v_{xx})_i^j - \frac{v_{i+1}^j - 2v_i^j + v_{i-1}^j}{(h_x)^2} \right]\end{aligned}\quad (3.2)$$

Using the Taylors Theorem.

$$\tau_i^j = -\frac{h_t}{2}(v_{tt})_i^j + \frac{c^2 h_x^2}{12}(v_{xxxx})_i^j \quad (3.3)$$

The local error is often used in place of the local discretization error.

Definition 3.1.2. *The local error is the difference between solutions of the partial differential equation and its finite difference approximation (i.e. $u_i^{j+1} - U_i^{j+1}$) assuming that no errors were committed prior to time level $j+1$. The finite difference approximation of the heat equation is*

$$\frac{U_i^{j+1} - U_i^j}{h_t} - c^2 \frac{U_{i+1}^j - 2U_i^j + U_{i-1}^j}{(h_x)^2} = 0$$

According to Definition (3.1) we commit no errors prior to time level $j+1$; thus, $U_i^j = u_i^j$ and

$$\frac{U_i^{j+1} - u_i^j}{h_t} - c^2 \frac{u_{i+1}^j - 2u_i^j + u_{i-1}^j}{(h_x)^2} = 0$$

Using (3.2) with v replaced by u and the heat equation reveals

$$\tau_i^j = -\frac{u_i^{j+1} - u_i^j}{h_t} + c^2 \frac{u_{i+1}^j - 2u_i^j + u_{i-1}^j}{(h_x)^2}$$

Combining the previous two relations, we obtain

$$u_i^{j+1} - U_i^{j+1} = -h_t \tau_i^j \quad (3.4)$$

Thus, the local error is the negative of the product of the local discretization error and the time step. This relationship between local and local discretization errors generally holds for explicit finite difference schemes.

Definition 3.1.3. *A finite difference scheme $P_\Delta U_i^j = 0$ is consistent with a partial differential equation $Pu = 0$ if the local discretization error tends to zero as $h_x \rightarrow 0$ and $h_t \rightarrow 0$.*

Equation (3.3) is the local discretization error for heat equation (1.1) i.e

$$\tau_i^j = -\frac{h_t}{2}(v_{tt})_i^j + \frac{c^2 h_x^2}{12}(v_{xxxx})_i^j$$

The left-hand side of this equation will tends to zero as h_t and h_x tend to zero. This means that the numerical scheme (2.8) tends to the exact equation at point x_i and at a time level t_j and therefore this approximation is consistent.

3.2 Stability

One of the central questions that arise by numerical treatment of the problem in question is stability of the numerical scheme. An algorithm for solving an evolutionary partial differential equation is said to be stable, if the numerical solution at a fixed time remains bounded as the step size goes to zero, so the perturbations in form of rounding error does not increase in time.

Definition 3.2.1. *A finite difference schemes for homogenous initial value problem is stable if there exists a constant value c independent of the mesh spacing and initial data such that*

$$\|U^n\| \leq C\|U^0\| \quad \text{as } n \rightarrow \infty, h_x \rightarrow 0 \quad \text{and} \quad h_t \rightarrow 0 \quad nh_t \leq T \quad (3.5)$$

Definition 3.2.2. *If we define the error to be the difference between the computed solutions and the exact solution of the discrete approximation, then the scheme is stable if the error remains uniformly bounded for successive iterations.*

Stability analysis

There are no general methods to verify the numerical stability for the partial differential equations in general form, so one restrict oneself to the case of linear Partial differential equation.

one of the methods used to analyse the numerical stability for partial differential equation is CFL condition.

3.2.1 CFL condition

In 1928 Richard Courant and Kurt Friedrichs and Hans Lewy, of the University of Göttingen in Germany, published a famous paper entitled "On the partial difference equations of mathematical physics". This paper was written long before the invention of digital computers, and its purpose in investigating finite difference approximations was to apply them to prove existence of solutions to partial differential equations. But the "CFL" paper laid the theoretical foundations for practical finite difference computations "too" and in particular, it identified a fundamental necessary condition for convergence of any numerical scheme that has subsequently come to be known as the CFL condition.

The CFL condition: For each (x, t) the mathematical domain of dependence is contained in the numerical domain of dependence

$$X(x, t) \subseteq X_0(x, t)$$

Theorem 3.2.1. *The CFL condition is a necessary condition for the convergence of a numerical approximation of a partial differential equation.*

The CFL Condition: A numerical method can be convergent only if its numerical domain of dependence contains the true domain of dependence of the Partial differential equation, at least in the limit as h_x and h_t go to zero.

For the Lax-Friedrichs, methods the condition on h_x and h_t required by the CFL condition is exactly the stability restriction. But it is important to note that in general the CFL condition is only a necessary condition. If it is violated then the method cannot be convergent. If it is satisfied, then the method might be convergent, but a proper stability analysis is required to prove this or to determine the proper stability restriction on h_x and h_t . The standard method for linear Partial differential equation with periodic boundary conditions was proposed by John von Neumann and is based on the representation of the rounding error in form of the Fourier series.

3.2.2 Von Neumann stability analysis

A scheme is stable if round off errors are not amplified in the calculations. The Fourier method can be used to check if a scheme is stable.

We used the forward difference for U_t and the centered difference for U_{xx} in the heat equation to arrive at the following difference equation

$$\frac{U_i^{(j+1)} - U_i^j}{\Delta t} = C^2 \frac{U_{(i+1)}^j - 2U_i^j + U_{(i-1)}^j}{(\Delta x)^2} \quad (3.6)$$

To better understand the effect of λ on the outcome of the scheme, let us consider a separated solution of the difference equation (3.6)

$$U_i^j = X_i T_j.$$

Substituting this solution gives

$$X_i T_{(j+1)} = \lambda (X_{(i+1)} T_j + X_{(i-1)} T_j) + (1 - 2\lambda) X_i T_j$$

Dividing both sides of the above equation by U_i^j , we obtain

$$\frac{T_{(j+1)}}{T_j} = (1 - 2\lambda) + \lambda \frac{X_{(i+1)} + X_{(i-1)}}{X_i} = \tau \quad (3.7)$$

where τ is independent of both i and j . This is a special case of linear difference equation of first order ;

$$T_{n+1} = a_n T_n + b_n$$

If T_o is already known it can be shown that T_n can be computed from T_o by the recipe

$$T_n = (\prod_{i=0}^{n-1} a_i) T_o + \sum_{r=0}^{n-1} (\prod_{i=r+1}^{n-1} a_i) b_r$$

In particular, if the difference equation is homogenous ($b_n \equiv 0$) and $a_n = \tau(\text{constant})$ then

$$T_n = \tau^n T_o \quad (3.8)$$

For the scheme (2.8) to be stable, we need to have $|\tau| \leq 1$, since otherwise it would lead to solutions that grow exponentially in time. The equation for the X component is

$$(1 - 2\lambda) + \lambda \frac{X_{(i+1)} + X_{(i-1)}}{X_i} = \tau$$

Plugging in a discretized Fourier mode $X_i = e^{jk_i h_x}$ into this equation, we obtain

$$\tau = (1 - 2\lambda) + \lambda \frac{e^{jk(i+1)h_x} + e^{jk(i-1)h_x}}{e^{jk_i h_x}}$$

or

$$\tau = 1 - 2\lambda + \lambda(e^{jkh_x} + e^{-jkh_x})$$

Using Euler's formula, we can rewrite the last identity as

$$\tau = 1 - 2\lambda + 2\lambda \cos(kh_x) = 1 - 2\lambda(1 - \cos(kh_x))$$

Since $\cos(kh_x) \leq 1$, we see that $\tau \leq 1$ for all values of $\lambda(\lambda > 0)$. On the other hand, the condition $-1 \leq \tau$ is equivalent to

$$-1 \leq 1 - 2\lambda(1 - \cos(kh_x))$$

The maximum case happens when the frequency k is such that $\cos(kh_x) \approx 1$, on the other hand the minimum case happens if $\cos kh_x \approx -1$ then the above inequality would give $-1 \leq 1 - 4\lambda$, which implies the following necessary condition for stability

$$c^2 \frac{h_t}{(h_x)^2} = \lambda \leq \frac{1}{2}$$

Which is bounded is usually taken as the stability condition of the finite-difference scheme (2.8). This means that if $\lambda \leq \frac{1}{2}$, then all round of errors will eventually decay, and the scheme is stable. The corresponding numerical solution will converge to the solution of initial boundary value problem. If, on the other hand, $\lambda > \frac{1}{2}$, then the errors will grow, thereby making the scheme unstable. Therefore it is only stable when $0 < \lambda \leq \frac{1}{2}$.

3.3 Convergence

Definition 3.3.1. *It means that the solution to a finite difference equation approaches the true solution to the partial differential equation as both grid interval and time step sizes are reduced.*

Theorem 3.3.1. (The Lax- Richtmyer Equivalence Theorem). *A consistent finite difference scheme for a partial differential equation for which the initial value problem is well posed is convergent if and only if it is stable.*

Proof. We will prove that a consistent and stable difference scheme is convergent
(consistency + stability $\Rightarrow$ convergence).

Let U^j be the solution of the difference equation from matrix equation 2.14 and let τ^j be the local discretization error. Then

$$u^{j+1} = Au^j + h_t \tau^j$$

Let $e^j = u^j - U^j$ be the global discretization error and subtract 2.14 to obtain

$$e^{j+1} = Ae^j + h_t \tau^j$$

For simplicity , assume that A is independent of j and iterate the above relation to obtain

$$e^j = A^j e^0 + h_t [A^{j-1} \tau^0 + A^{j-2} \tau^1 + A^{j-3} \tau^2 + \dots + \tau^{j-1}]$$

Taking a norm and noting that $e^0 = 0$ in the absence of round-off error,

$$\|e^j\| \leq h_t [\|A^{j-1}\| \|\tau^0\| + \|A^{j-2}\| \|\tau^1\| + \|A^{j-3}\| \|\tau^2\| + \dots + \|\tau^{j-1}\|] \quad (3.9)$$

The difference scheme is consistent, we can make τ^k arbitrary small by making h_x and h_t sufficiently very small. Let us choose a total time interest T and $\epsilon > 0$ such that $\|\tau^k\| < \epsilon$ for all $kh_t \leq T$ whenever $h_t < \delta$. Furthermore the scheme is stable , there exists a constant C such that $\|A^k\| \leq C$ for all $kh_t \leq T$ with the above Equation 2.14 , these consideration implies that

$$\|e^j\| \leq j h_t C \epsilon$$

If $j h_t \leq T$

$$\|e^j\| = \|u^j - U^j\| \leq TC\epsilon$$

and the scheme converges.

□

Now that we have discussed about consistence and stability. It is time to see how they are connected. As we have stated in the Lax Equivalence Theorem.

This proves that under the condition, $0 < \lambda \leq \frac{1}{2}$ of explicit scheme equation (2.8) is stable and it is thus convergent, thanks to the Lax equivalent theorem.

Chapter 4

Implementation

Matlab implementation of the explicit finite difference scheme of heat equation of insulated end is presented and demonstrated in this section.

4.1 Test Problem

Consider the initial and boundary problem of (1.1)

$$\begin{cases} U_t = c^2 U_{xx} & 0 < x < L, t > 0 \\ U_x(0, t) = U_x(L, t) = 0, & t > 0 \\ U(x, 0) = f(x), & 0 < x < L. \end{cases} \quad (4.1)$$

Using method of separation of variables the exact analytical solution of the above heat equation is:

$$u(x, t) = a_0 + \sum_{n=1}^{\infty} a_n e^{-\beta_n^2 t} \cos \frac{n\pi}{L} x \quad (4.2)$$

where

$$\begin{aligned} a_0 &= \frac{1}{L} \int_0^L f(x) dx \\ a_n &= \frac{2}{L} \int_0^L f(x) \cos \frac{n\pi}{L} x dx \\ \beta_n &= c \frac{n\pi}{L} \quad n = 1, 2, \dots \end{aligned}$$

Use the explicit scheme (2.8) to simulate the following example.

Example 4.1.1. *Test the result of the explicit finite difference scheme (2.8) using the following data:*

$$c = L = 1 \text{ and } f(x) = \cos \frac{2\pi x}{L}.$$

Using the above data and by (4.2) the exact solution of the heat equation is:

$$u(x, t) = e^{-(2\pi)^2 t} \cos 2\pi x \quad (4.3)$$

4.2 Implementation

The explicit finite difference scheme (2.8) is implemented in the Matlab function *heat* as described here under in listing 1.

```
function [xo,to,Uo] = heat(nt,nx,alpha,L,tmax)
% Solve 1D heat equation with the finite difference scheme
% Synopsis: heat
% heat(nt)
% heat(nt,nx)
% heat(nt,nx,alpha)
% heat(nt,nx,alpha,L)
% heat(nt,nx,alpha,L,tmax)
% Input: nt = number of time steps.
% nx = number of mesh points in x direction.
% alpha = the rate of diffusion.
% L = length of the domain.
% tmax = maximum time for the simulation.
% Output: x = location of finite difference nodes
% t = values of time at which solution is obtained (time nodes)
% U = matrix of solutions: U(:,j) is U(x) at t = t(j)
if nargin<1, nt =101; end
if nargin<2, nx =11; end
if nargin<3, alpha = 0.1; end
if nargin<4, L = 1; end
if nargin<5, tmax = 0.1; end
% --- Compute mesh spacing and time step
dx = L/(nx-1);
dt = tmax/(nt-1);
r = alpha*dt/dx^2; r2 = 1 - 2*r;
% --- Create arrays to save data for export
x = linspace(0,L,nx)';
t = linspace(0,tmax,nt);
U = zeros(nx,nt);
% --- Set IC and BC
U(:,1) = cos(2*pi*x/L);
% --- Loop over time steps
for m=2:nt
    U(1,m) = r2*U(1,m-1) + 2*r*U(2,m-1);
end
    for m=2:nt
        U(11,m) = r2*U(11,m-1) + 2*r*U(10,m-1);
    end
    for m=2:nt
        for i=2:nx-1
            U(i,m) = r*U(i-1,m-1) + r2*U(i,m-1) + r*U(i+1,m-1);
        end
    end
% --- Compare with exact solution at end of simulation
ue = cos(2*pi*x/L)*exp(-t*alpha*(2*pi/L)^2);
plot(x,ue(:,8),'r--',x,U(:,8),'b--','LineWidth',2);
    title('Heat');
    xlabel('x','FontSize',14);
    ylabel('U(x,t)','FontSize',14);
    legend('exact','appro');
```

Listing 1: Code for Matlab implementation of the explicit finite difference scheme for solution to the heat equation.

Running *heat* with the above parameters gives

Case 1. For $r = \lambda = c^2 \frac{h_t}{(h_x)^2} \leq \frac{1}{2}$

Let $r = \lambda = \frac{1}{1000}$, then the exact and approximate solutions are shown below

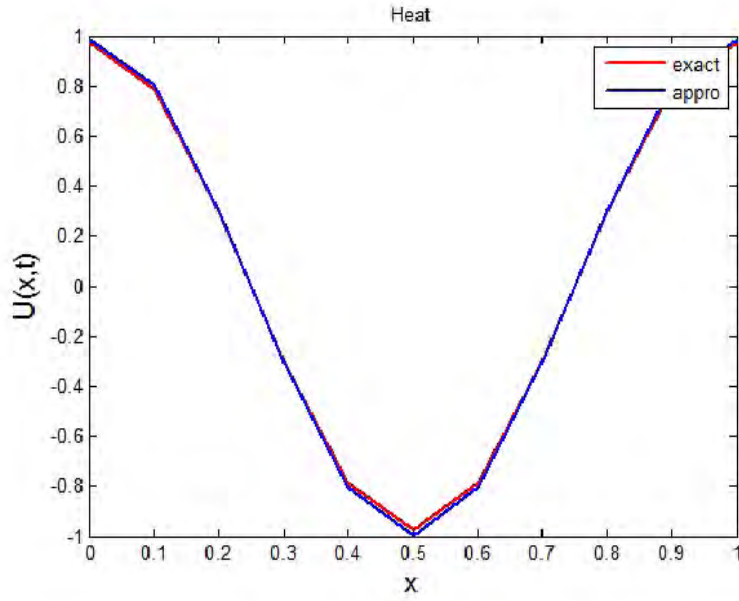


Figure 1: The explicit finite difference scheme solution to the heat equation at the 8th time level obtained with $\lambda = \frac{1}{1000}$. The solution is stable.

The explicit finite difference scheme (2.8) is implemented in the Matlab function *heat* as described here under in listing 2.

```
function [xo,to,Uo] = heat (nt,nx,alpha,L,tmax)
% Solve 1D heat equation with the finite difference scheme
% Synopsis: heat
% heat(nt)
% heat(nt,nx)
% heat(nt,nx,alpha)
% heat(nt,nx,alpha,L)
% heat(nt,nx,alpha,L,tmax)
% Input: nt = number of steps.
% nx = number of mesh points in x direction.
% alpha = the rate of diffusion.
% L = length of the domain.
% tmax = maximum time for the simulation.
% Output: x = location of finite difference nodes
% t = values of time at which solution is obtained (time nodes)
% U = matrix of solutions: U(:,j) is U(x) at t = t(j)
if nargin<1, nt =101; end
if nargin<2, nx =11; end
if nargin<3, alpha = 1; end
if nargin<4, L = 1; end
if nargin<5, tmax = 0.65; end
% --- Compute mesh spacing and time step
dx = L/(nx-1);
dt = tmax/(nt-1);
r = alpha^2*dt/dx^2; r2 = 1 - 2*r;
% --- Create arrays to save data for export
x = linspace(0,L,nx)';
t = linspace(0,tmax,nt);
U = zeros(nx,nt);
% --- Set IC and BC
U(:,1) = cos(2*pi*x/L);
%U(:,2) = sin(2*pi*x/L) - 2*(pi/L)^2*dt^2*sin(2*pi*x/L);% implies u0 = 0;uL = 0;
%u0 = 0; uL = 0; % needed to apply BC inside time step loop
% --- Loop over time steps
for m=2:nt
    U(1,m) = r2*U(1,m-1) + 2*r*U(2,m-1);
end
for m=2:nt
    U(11,m) = r2*U(11,m-1) + 2*r*U(10,m-1);
end
for m=2:nt
    for i=2:nx-1
        U(i,m) = r*U(i-1,m-1) + r2*U(i,m-1) + r*U(i+1,m-1);
    end
end
% --- Compare with exact solution at end of simulation
ue = cos(2*pi*x/L)*exp(-t*alpha*(2*pi/L)^2);

plot(x,ue(:,8),'r--',x,U(:,8),'b--','LineWidth',2);
    title('Heat');
    xlabel('x','FontSize',14);
    ylabel('U(x,t)','FontSize',14);
    legend('exact','appro');
```

Listing 2: Code for Matlab implementation of the explicit finite difference scheme for solution to the heat equation.

Case 2. For $r = \lambda = c^2 \frac{h_t}{(h_x)^2} > \frac{1}{2}$
Let $r = \lambda = \frac{13}{20}$, then the exact and approximate solutions are shown below

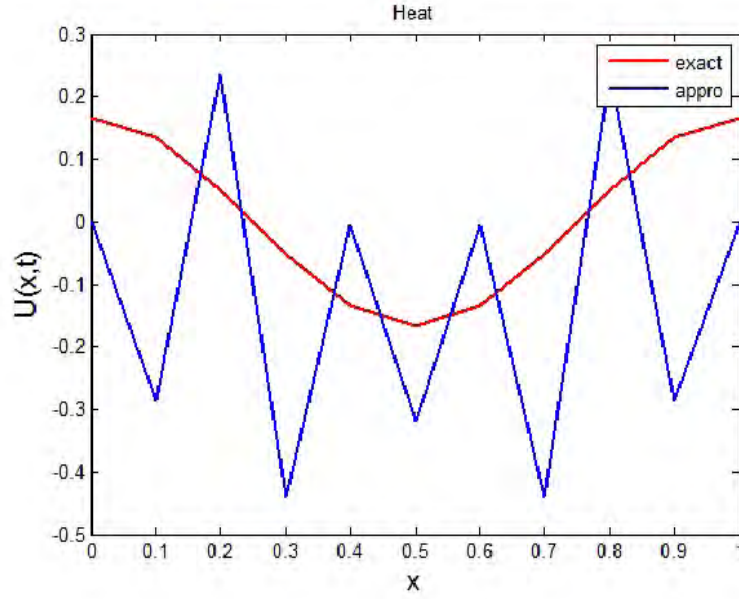


Figure 2: The explicit finite difference scheme solution to the heat equation at the 8th obtained with $\lambda = \frac{13}{20}$. The solution is unstable. We can see from the Figures 1 and 2 that the numerical solution is stable when $\lambda \leq \frac{1}{2}$ and is unstable when $\lambda > \frac{1}{2}$. The explicit scheme (2.8) is conditional stable.

Summary

Using the forward difference approximation for the time derivative in the heat equation, we obtained the explicit scheme (2.8) and the stability analysis of these schemes showed that for stability of the first scheme we need $h_t \leq \frac{1}{2}(h_x)^2$. This means that if $\lambda \leq \frac{1}{2}$, then all round of errors will eventually decay, and the scheme is stable. The corresponding numerical solution will converge to the solution of initial boundary value problem. If, on the other hand, $\lambda > \frac{1}{2}$, then the errors will grow, thereby making the scheme unstable. Therefore it is only stable when $0 < \lambda \leq \frac{1}{2}$.

Bibliography

- [1] Randall J. LeVeque, *Finite Difference Methods for Ordinary and Partial Differential Equations*, Philadelphia, Washington, 2007.
- [2] J.W. Thomas., *Numerical partial Equations: Finite Difference Methods*, Springer-Verlag, New York, 1995.
- [3] K.W. Marton and David Mayers, *Numerical Solution of Partial Differential Equations*, Cambridge University Press, New York, second edition, 2005.
- [4] R. Haberman, *Elementary Applied Partial Differential Equations with Fourier series and Boundary value problems*, Prentices Hall, Second edition, 1993
- [5] William F. Ames, *Numerical Methods for Partial Differential Equations*, Academic Press, Inc., Boston, third edition, 1992.
- [6] A.Coco, G.currenti, C.DelNegro and G.Russo, *A second order Finite - Difference Ghost point Method for Elasticity problem on unbounded domains with application to volcanology*, *commun. comput.phys.*(2013).
- [7] A.Coco and G.Russo, *Finite-Difference ghost-point Multigrid methods on cartesian gride for elliptic problem in arbitrary domains*, *Journal of comput.phys.*(2013)