FINITE DIFFERENCE SCHEMES ARISING FROM OPERATOR SPLITTING FOR SOLVING TWO DIMENSIONAL SYSTEM OF BURGERS' EQUATION

BY

ROTICH JOHN KIMUTAI

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN APPLIED MATHEMATICS OF

UNIVERSITY OF ELDORET, KENYA.

FEBRUARY, 2016

DECLARATION

DECLARATION BY THE CANDIDATE:

This thesis is my original work and has not been submitted for any academic award in any institution; and shall not be reproduced in part or full, or in any format without prior written permission from the author and/or University of Eldoret.

Rotich John Kimutai

Signature:_____

Date:_____

Reg. No.: SC/D.PHIL/007/10

DECLARATION BY SUPERVISORS:

This thesis has been submitted with our approval as University Supervisors.

Prof. J. K. Bitok

Signature:_____

Date:_____

University of Eldoret

Dr. M. Z. Mapelu

Signature:_____

Date:_____

University of Eldoret

DEDICATION

I dedicate this work to my wife Rose for her love, patience, encouragement, support and prayers. Not forgetting to my children Lauryn and Victor, my parents: Andrew and Rael, my brothers, sisters and in-laws and other relatives and friends who gave me moral support throughout my study.

ABSTRACT

Solving Burgers equation continues to be a challenging problem. Burgers' equation is a fundamental partial differential equation from fluid mechanics. It occurs in various areas of applied mathematics, such as modeling of fluid dynamics and traffic flow. It relates to the Navier-Stokes equation for incompressible flow with the pressure term removed. So far the methods that have been used to solve such equations are: Alternative Direction Implicit (ADI) methods, Variation of Iteration Method (VIM), locally one dimensional method and Finite Difference Method (FDM) which is used in this work. The study developed the pure Crank-Nicholson (CN), Crank-Nicholson-Du-Fort and Frankel (CN-DF), Crank-Nicholson- Lax-Friedrichs'(CN-LF) and Crank-Nicholson- Du-Fort and Frankel-Lax-Friedrichs' (CN-DF-LF) schemes by Operator Splitting. Crank-Nicholson-Du-Fort and Frankel is an hybrid scheme made by combining the Crank-Nicholson and Du-Fort and Frankel schemes which are both unconditionally stable but the Du-fort scheme is explicit while the Crank-Nicholson scheme is implicit and the Crank-Nicholson-Lax-Friedrichs' scheme is a hybrid scheme made up of combining the Crank-Nicholson and Lax-Friedrichs' scheme. Lax-Friedrichs' scheme is conditionally stable and an explicit scheme while the Crank-Nicholson- Du-Fort and Frankel-Lax-Friedrichs' method is a hybrid scheme made by combining the Crank-Nicholson, Du-Fort and Frankel and Lax-Friedrichs' schemes. Crank-Nicholson-Du-Fort and Frankel is an hybrid scheme made by combining the Crank-Nicholson and Du-Fort and Frankel schemes which are both unconditionally stable but the Du-fort scheme is explicit while the Crank-Nicholson scheme is implicit. The developed schemes were solved numerically using MATLAB was used to generate the results. Analysis of the schemes showed that they are consistent, convergent and stable.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS, ACRONYMS AND SYMBOLS

ADI	–	Alternating Direction Implicit methods
BTCS	–	Backward in Time Central in Space
CN	–	Crank-Nicholson
CN-DF	–	Crank-Nicholson- Du-Fort and Frankel
CN-LF	–	Crank-Nicholson-Lax-Friedrich's
CN-DF-LF	–	Crank-Nicholson- Du-Fort and Frankel- Lax-Friedrich's
FD	–	Finite Difference
FTCS	–	Forward in Time Central in Space
MATLAB	–	Matrix Laboratory
Re	–	Reynolds number
2-D	–	Two Dimensional
3-D	–	Three Dimensional
Δ	–	Forward difference operator
∇	–	Backward difference operator
δ	–	Central difference operator
μ	–	Averaging operator
$\mathbb{R}$	–	Set of Real Numbers
$\mathbb{N}$	–	Set of Natural Numbers

ACKNOWLEDGEMENTS

First I am very grateful to our Almighty God for the life, care and provision. Also to all who have contributed in one way or the other towards the success of this work, most important is the late Dr. Koross, for guiding me in writing the proposal just before his demise. All members of staff in University of Eldoret, Mathematics and Computer Science department and especially Dr. Kimeli who is also the Dean of School of Science, Dr. Korir who is the head of department and Dr. Chepkwony, Dr. Manyonge, Dr. Musembi and Dr. Iyaya who participated in the taught courses and all my classmates Mr. Maritim, Mr. Wambua, Mr. Kirui, Mr. Cheres, Mr. Okong'o and Mr. Limo for helping me out in one way or the other.

Thanks to the staff of University of Kabianga especially the Mathematics and Computer Science department, where I work, for assisting me with ideas and being there to give moral support and advice.

Am grateful to the staff and the Administration of 'National Council of Science and Technology' (NACOSTI) and University of Kabianga for funding part of my research. Not forgetting the Pastors and all members of the Happy Church Ministries International, especially those from Kericho region and the overseer, Kericho region, Rev. Reuben Kemei of Kipsolu Happy Church for prayers and encouragement throughout the research.

May the Lord bless you all.

Thank you!

CHAPTER ONE

INTRODUCTION

1.1 Introduction

In this chapter the study introduces partial differential equations, non-linear parabolic equations and the two-dimensional Burgers equation. Some basic concepts are defined and the problem that the thesis is handling is given. The study also states the objectives of the research and discusses the justification of the research.

1.2 Background Information

Partial differential equations are applied frequently in science, engineering and mathematics. Many partial differential equations cannot be solved by analytical methods in closed form solution. In most research work in fields like: applied elasticity, theory of plate and shells, hydro-dynamics, quantum mechanics among others, the research problems reduce to partial differential equations. Since analytic solutions are not available, numerical solutions of the partial differential equations by various methods is used. Certain types of boundary value problems can be solved by replacing the differential equation by the corresponding finite difference equation and then solving by a process of iteration. These methods have been used by many mathematicians (Jain, 2004). Linearized parabolic equations appear as models in heat flow and gas dynamics. Finite difference solutions of these equations are found by using ordinary discretization (Ames, 1992; Mitchel & Grffiths, 1980)). These methods give fairly accurate results.

1.3 Non-Linear Parabolic Equation

An n-dimensional non-linear parabolic equation is of the form:

$$\vec{u}_t = F\left(t, \vec{X}, \vec{u}, \frac{\partial^r \vec{u}}{\partial x^r}\right), \Omega \times (t \geq 0) \qquad (1.1)$$

$$\vec{u}(\vec{X},0) = u_0(\vec{X}), \vec{u} = (u,v) \tag{1.2}$$

where $\vec{X} = (x_1, x_2, \dots x_n)^T$ is the space vector, $\frac{\partial^r \vec{u}}{\partial x^r}$ is the r^{th} order partial derivative of $\vec{u}$ with respect to x^r and $\Omega \in \Re^n$ where Ω is the boundary for $\vec{X}$.

1.4 Two Dimensional Burgers' Equation

The 2-D Burgers' equation is an example of the non-linear parabolic equations and is defined as follows:

$$\left.\begin{aligned} f_1(u,v) = u_t = -uu_x - vu_y + \tfrac{1}{Re}(u_{xx} + u_{yy}) \\ f_2(u,v) = v_t = -uv_x - vv_y + \tfrac{1}{Re}(v_{xx} + v_{yy}) \end{aligned}\right\} \tag{1.3}$$

Subject to initial conditions:

$$\left.\begin{aligned} u(x,y,0) = f(x,y),\ (x,y) \in D \\ v(x,y,0) = g(x,y),\ (x,y) \in D \end{aligned}\right\} \tag{1.4}$$

and boundary conditions:

$$\left.\begin{aligned} u(x,y,t) = f_3(x,y),\ x,y \in \partial D, t > 0 \\ v(x,y,t) = g_1(x,y),\ x,y \in \partial D, t > 0 \end{aligned}\right\} \tag{1.5}$$

where $D = \{(x,y) | a \le x \le b, a \le y \le b\}$ and ∂D is its boundary $u(x,y,t)$ and $v(x,y,t)$ are the velocity components to be determined, f, g, f_1, f_2, f_3 and g_1 are known functions and Re is the Reynolds number.

Which is a fundamental partial differential equation in fluid mechanics and it occurs in various areas of applied mathematics, such as modeling of fluid dynamics, heat conduction, and acoustic waves (Hongqing *et al.*, 2010)

1.5 Basic Concepts

1.5.1 Theory of existence and uniqueness

Without loss of generality the existence and uniqueness theorem for second order parabolic equation is discussed.

Theorem 1.1 [uniqueness]

There exists at most one solution of the initial and or boundary value problem

$$u_t = u_{xx} + u_{yy} + f(x,t) \in \Omega \subseteq \Re^2, \Omega \subseteq (0 < X < x)(0 < Y < y)(0 < t \leq T) \tag{1.6}$$

with initial condition:

$$u(x, y, 0) = \phi(x, y) \tag{1.7}$$

and boundary condition:

$$\begin{aligned} &u(x, y, t) = g(x, y, t), \\ &u(x, y, t) \in \partial\Omega, 0 < t \leq T \end{aligned} \tag{1.8}$$

where Ω is bounded as shown above.

The proof of the theorem is given in (Levandosky, 2001)

1.5.2 Finite difference Approximations and their operators

Considering the case

$$R : (a \leq x, y \leq b)(t \geq 0). \tag{1.9}$$

In order to obtain a finite difference replacement of problem (1.6) the region R is assumed to be covered by rectilinear grid (mesh) with sides parallel to the $x-, y-$ and $t-axes$, with h, q and k being the grid spacing in the $x-, y-$ and $t-$directions respectively.

The grid points (x, y, t) are given by $x = mh, y = lq, t = nk$ where m, l and n are integers. $m = l = n = 0$ is the origin.

The idea of using finite difference to solve partial differential equations (P.D.E s) is to select a grid in time and space (with mesh lengths $h = \Delta x$, $q = \Delta y$ and $k = \Delta t$ respectively) and to approximate the values $U(mh, lq, nk)$. The functions satisfying the difference and differential equations at the mesh point $x = mh, y = lq, t = nk$ are denoted by $U_{m,l,n}$ and $u_{m,l,n}$ respectively.

If $u = u(x, y, t)$ is a function, its partial derivatives are approximated by:

1.5.3 Forward space difference

$$\Delta U(x,y,t) = \frac{U(x+h,y,t) - U(x,y,t)}{h} + \frac{U(x,y+q,t) - U(x,y,t)}{q}$$

$$= \frac{U_{m+1,l,n} - U_{m,l,n}}{h} + \frac{U_{m,l+1,n} - U_{m,l,n}}{q} \tag{1.10}$$

1.5.4 Forward time difference

$$\Delta U(x,t) = \frac{U(x,y,t+k) - U(x,y,t)}{k} = \frac{U_{m,l,n+1} - U_{m,l,n}}{k} \tag{1.11}$$

1.5.5 Backward space difference

$$\nabla U(x,t) = \frac{U(x,y,t) - U(x-h,y,t)}{h} + \frac{U(x,y,t) - U(x,y-q,t)}{q}$$

$$= \frac{U_{m,l,n} - U_{m-1,l,n}}{h} + \frac{U_{m,l,n} - U_{m,l-1,n}}{q} \tag{1.12}$$

1.5.6 Centered space difference

$$\delta U(x,y,t) = \frac{U(x+h,y,t)-U(x-h,y,t)}{2h} + \frac{U(x,y+q,t)-U(x,y-q,t)}{2q}$$

$$= \frac{U_{m+1,l,n}-U_{m-1,l,n}}{2h} + \frac{U_{m,l+1,n}-U_{m,l-1,n}}{2q} \tag{1.13}$$

In most cases the central difference operator δ is used and the first order central differences are given by

$$\delta_x U(x,y,t) = U(x+\tfrac{1}{2}h,y,t) - U(x-\tfrac{1}{2}h,y,t)$$

$$\delta_y U(x,y,t) = U(x,y+\tfrac{1}{2}q,t) - U(x,y-\tfrac{1}{2}q,t) \tag{1.14}$$

which in notation can be written as

$$\delta_x U_{m,l,n} = U_{m+\frac{1}{2},l,n} - U_{m-\frac{1}{2},l,n}$$

$$\delta_y U_{m,l,n} = U_{m,l+\frac{1}{2},n} - U_{m,l-\frac{1}{2},n} \tag{1.15}$$

and

$$\delta_t U(x,y,t) = U(x,y,t+\frac{1}{2}k) - U(x,y,t-\frac{1}{2}k) \tag{1.16}$$

that is

$$\delta_t U_{m,l,n} = U_{m,l,n+\frac{1}{2}} - U_{m,l,n-\frac{1}{2}} \tag{1.17}$$

$\delta U_{m,l,n}$ is usually approximated by

$\mu\delta U_{m,l,n}$ where μ is the averaging operator, thus

$$\mu\delta U_i = \frac{1}{2}(U_{i+1} + U_{i-1}) \tag{1.18}$$

The second order central space difference is given by

$$\delta_x^2 U_{m,l,n} = \delta_x(\delta_x U_{m,l,n}) = U_{m+1,l,n} - 2U_{m,l,n} + U_{m-1,l,n}$$

$$\delta_y^2 U_{m,l,n} = \delta_y(\delta_y U_{m,l,n}) = U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n} \tag{1.19}$$

and

$$\delta_x^3 U_{m,n} = \delta_x(\delta_x^2 U_{m,n}) = \delta_x(U_{m+1,l,n} - 2U_{m,l,n} + U_{m-1,l,n})$$

$$\delta_y^3 U_{m,n} = \delta_y(\delta_y^2 U_{m,n}) = \delta_y(U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n}) \tag{1.20}$$

In general

$$\delta_i^n U_{m,l,n} = \delta_i(\delta_i^{n-1} U_{m,l,n}),\ n \in Z^+ \text{ and } i \text{ can be } x, y \text{ or } t \tag{1.21}$$

1.5.7 Truncation Error

Let the solution of the partial differential equation (1.6) at the mesh point (mh, lq, nk) be $U_{m,l,n}$ using Taylors series

$$u_{m,l,n+1} = u(x_m, y_l, t_n + k) = u(x_{m,l}, t_n) + k\left(\frac{\partial u}{\partial t}\right)_{m,l,n} + \frac{k^2}{2!}\left(\frac{\partial^2 u}{\partial t^2}\right)_{m,l,n} + O(k^3) \tag{1.22}$$

or equivalently

$$u_{m,l,n+1} = u_{m,l,n} + k(u_t)_{m,l,n} + \frac{k^2}{2!}(u_{tt})_{m,l,n} + \frac{k^3}{3!}(u_{ttt})_{m,l,n} + O(k^4) \tag{1.23}$$

Also

$$u_{m+1,l,n} = u_{m,l,n} + h(u_x)_{m,l,n} + \frac{h^2}{2!}(u_{xx})_{m,l,n} + \frac{h^3}{3!}(u_{xxx})_{m,l,n} + O(h^4) \tag{1.24}$$

$$u_{m,l+1,n} = u_{m,l,n} + q(u_y)_{m,l,n} + \frac{q^2}{2!}(u_{yy})_{m,l,n} + \frac{q^3}{3!}(u_{yyy})_{m,l,n} + O(q^4) \tag{1.25}$$

and

$$u_{m-1,l,n} = u_{m,l,n} - h(u_x)_{m,l,n} + \frac{h^2}{2!}(u_{xx})_{m,l,n} - \frac{h^3}{3!}(u_{xxx})_{m,l,n} + \frac{h^4}{4!}(u_{xxxx})_{m,l,n} + O(h^5) \tag{1.26}$$

$$u_{m,l-1,n} = u_{m,l,n} - q(u_y)_{m,l,n} + \frac{q^2}{2!}(u_{yy})_{m,l,n} - \frac{q^3}{3!}(u_{yyy})_{m,l,n} + \frac{q^4}{4!}(u_{yyyy})_{m,l,n} + O(q^5) \tag{1.27}$$

In general

$$u_{m,l,n+1} = \sum_{i=0}^{\vartheta-1} \frac{k^i}{i!}(u_{t^i})_{m,l,n} + O(k^{\vartheta}) \tag{1.28}$$

where u_{tt} is the second order partial derivative of u with respect to t. This is said to be accurate to order ϑ. Similarly

$$u_{m+1,l,n} = \sum_{j=0}^{\eta-1} \frac{k^j}{j!}\left(u_{jx}\right)_{m,l,n} + O(k^{\eta}) \tag{1.29}$$

The truncation error of the finite difference scheme is given by

$$T = H_t\left(U_{m,l,n}\right) - F_x\left(U_{m,l,n}\right) - F_y(U_{m,l,n}) \tag{1.30}$$

where $H_t\left(U_{m,l,n}\right)$ is either

$$\frac{1}{k}\Delta_t(U_{m,l,n}) \text{ or } \frac{1}{k}\nabla_t(U_{m,l,n}) \text{ or } \frac{1}{2k}\mu\delta_t(U_{m,l,n})$$

and

$F_x\left(U_{m,l,n}\right)$ and $F_y(U_{m,l,n})$ are the spatial discretization which can take any form according to our desire. Now the study illustrate truncation error by use of an example. For the sample parabolic equation

$$u_t = u_{xx} + u_{yy} \tag{1.31}$$

the explicit ordinary finite difference method of solving it is given by

$$\frac{U_{m,l,n+1} - U_{m,l,n}}{k} = \frac{U_{m+1,l,n} - 2U_{m,l,n} + U_{m-1,l,n}}{2h} + \frac{U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n}}{2q} \tag{1.32}$$

$$T = \frac{U_{m,l,n+1} - U_{m,l,n}}{k} - \frac{U_{m+1,l,n} - 2U_{m,l,n} + U_{m-1,l,n}}{2h} - \frac{U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n}}{2q} \tag{1.33}$$

Using the Taylor's series (1.23) to (1.27) in (1.33) to get

$$T = \frac{1}{2}ku_{tt} - \frac{1}{12}h^2u_{xxxx} - \frac{1}{12}q^2u_{yyyy} + \text{.....} \tag{1.34}$$

and so equation (1.30) can be written as

$$\frac{U_{m,l,n+1}-U_{m,l,n}}{k}=\frac{U_{m,l,n}-2U_{m,l,n}+U_{m-1,l,n}}{h^2}+\frac{U_{m,l+1,n}-2U_{m,l,n}+U_{m,l-1,n}}{q^2}+O(k+h^2+q^2)\,.$$

(1.35)

T is called the truncation error and $O(k+h^2+q^2)$ in the above equation (1.35) is called the order of the truncation error.

Definition 1.1[consistency]

A finite difference scheme

$$P_{k,h,q}U=f \tag{1.36}$$

is consistent with partial differential equation

$$Pu=f \tag{1.37}$$

of order (r,s,m) if for any smooth function ϕ,

$$P\phi-P_{k,h,q}\phi=O(k^r,h^s,q^m) \tag{1.38}$$

Definition 1.2 [L^2 norm].

For a function $(w=l,.....,w_{-3},w_{-2},w_{-1},w_0,w_1,w_2,w_3...)$ on a grid with step size h:

$$\|w\|=\left(h\sum_{m=-\infty}^{\infty}|w_m|^2\right)^{\frac{1}{2}}. \tag{1.39}$$

For a function f on the real line;

$$\|f\|=\left(\int_{-\infty}^{\infty}|f(x)|^2dx\right)^{\frac{1}{2}} \tag{1.40}$$

Definition 1.3 [convergence]**.**

A one-step finite difference scheme approximating a partial differential equation is called convergent if for any solution to the partial differential equation $u(x,t)$, and the solution to the finite difference scheme $U_{m,l,n}$ is such that $U_{m,l,0} \to u(x,y,0)$ as

$mh \to x$, and $lq \to y$ leads to:

$U_{m,l,n} \to u(x,y,t)$ as $(mh, lq, nk) \to (x,y,t)$ as $h, q, k \to 0$.

1.5.8 Some second order finite difference schemes

Consider the simple parabolic equation;

$$u_t = \lambda(u_{xx} + u_{yy}). \tag{1.41}$$

where λ is the diffusion coefficient.

Several finite difference schemes for solving the simple diffusion equation (1.41) are given by several authors including Jain (2004), Morton and Mayers (2005), Rahman (1998) among others. These authors discuss simple difference schemes due to Schmidt, Crank-Nicholson, Du-Fort and Frankel and three level formulae. The Schmidt difference method for solving equation (1.41) is given by

$$U_{m,l,n+1} = U_{m,l,n} + \lambda r(U_{m+1,l,n} - 2U_{m,l,n} + U_{m-1,l,n} + U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n}) \tag{1.42}$$

where $r = \frac{k}{h^2}$ for $h = q$

k, h and q are time and spatial stepping. This is the simplest explicit finite difference scheme.

The following are simple finite difference schemes for finding its numerical solution.

1.5.9 Forward-Time Centered- Space (FTCS) Scheme

The discretization of the equation is:

$$\Delta_t U(x,y,t) = \frac{1}{h^2}\delta_x^2 U(x,y,t) + \frac{1}{q^2}\delta_y^2 U(x,y,t) \tag{1.43}$$

which can be written as

$$\frac{U_{m,l,n+1} - U_{m,l,n}}{k} = \frac{U_{m+1,l,n} - 2U_{m,l,n} + U_{m-1,l,n}}{h^2} + \frac{U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n}}{q^2} + O(k + h^2 + q^2) \tag{1.44}$$

1.5.10 Lax- Friedrich's Scheme

In (1.43) the term $U_{m,l,n}$ on the left hand side is replaced by

$$\frac{1}{2}(U_{m+1,l,n} + U_{m-1,l,n})$$

Thus obtaining

$$\frac{U_{m,l,n+1} - \frac{1}{2}(U_{m+1,l,n} + U_{m-1,l,n})}{k} = \frac{U_{m+1,l,n} - (U_{m+1,l,n} + U_{m-1,l,n}) + U_{m-1,l,n}}{h^2} + \frac{U_{m,l+1,n} - (U_{m,l+1,n} + U_{m,l-1,n}) + U_{m,l-1,n}}{q^2} + O(k + h^2 + q^2) \tag{1.45}$$

1.5.11 Leap-Frog Scheme

Discretization is centered in both time and space (CTCS), i.e

$$\Delta_t U(x,y,t) = \frac{1}{h^2}\delta_x^2 U(x,y,t) + \frac{1}{q^2}\delta_y^2 U(x,y,t) \tag{1.46}$$

thus obtaining

$$\frac{U_{m,l,n+1}-U_{m,l,n-1}}{2k}=\frac{U_{m+1,l,n}-2U_{m,l,n}+U_{m-1,l,n}}{h^2}+\frac{U_{m,l+1,n}-2U_{m,l,n}+U_{m,l-1,n}}{q^2}+O(k^2+h^2+q^2) \tag{1.47}$$

1.5.12 Crank-Nicholson Scheme

Discretization in this case is forward in time and central in space but two points are considered (mh, lq, nk) and $(mh, lq, (n+1)k)$

The scheme is given by:

$$\frac{U_{m,l,n+1}-U_{m,l,n}}{k}=\frac{1}{h^2}\delta_x^2(U_{m,l,n+1}+U_{m,l,n})+\frac{1}{q^2}\delta_y^2(U_{m,l,n+1}+U_{m,l,n})+O(k^2+h^4+q^4) \tag{1.48}$$

1.5.13 Du-Fort and Frankel Scheme

In the right hand side of equation (1.48) the term $U_{m,l,n}$ is replaced by $\frac{1}{2}(U_{m,l,n-1}+U_{m,l,n+1})$ thus obtaining:

$$\frac{U_{m,n+1}-U_{m,n-1}}{2k}=\frac{U_{m+1,n}-(U_{m,n+1}+U_{m,n-1})+U_{m-1,n}}{h^2}+\frac{U_{m,l+1,n}-(U_{m,l,n+1}+U_{m,n-1})+U_{m,l-1,n}}{h^2}+ O(k^2+h^2+q^2) \tag{1.49}$$

1.5.14 Operator Splitting Methods

Operator splitting methods are well known in the field of numerical solution of partial differential equations. The technique is generally used in one of the two ways: It is used in methods in which one splits the differential operator such that each split system only involves derivatives along one of the coordinate axes. Alternatively, it is used as a means to split the differential operator into several parts, where each part represents a particular physical phenomenon, such as heat transfer, convection, diffusion, among others. In all these cases, the corresponding numerical method is defined as a sequence of solves of

each of the split problems. This can lead to very efficient methods, since one can treat each part of the original operator independently.

Operator splitting means the spatial differential operator appearing in the equations is split into a sum of different sub-operators having simpler forms, and the corresponding equations can be solved easier. Operator splitting is an attractive technique for solving coupled systems of partial differential equations, since complex equation system may be split into simpler parts that are easier to solve. Several operator splitting techniques exists.

In realistic applications the operators corresponds to physical operators such as convection and diffusion operators.

Splitting methods assume that the mathematical problem can be split into two or more terms.

While attractive from a theoretical point of view, the fractional operator splitting methods based on exact flows may not be practically feasible. In particular, the exponential mapping may not be computationally available or too expensive to evaluate exactly.

Thus the flow map experiment is often approximated using some numerical method. Some of the choices studied in the literature are regular ODE-based integration of a single component of the vector field. A feature of numerical approximations to the exponential function is that such approximations usually do not satisfy the composition property experienced by the exact flow. Distinguishing the different approaches, methods based on exact flows are commonly known as exponential splitting methods.

Having constructed splitting methods for ordinary differential equations, the question naturally arises of how to construct accurate schemes which may be used with non-small step size.

The splitting method is one of the most powerful method to solve the abstract Cauchy problems. The main idea is to lead the complex problem to the sequence of sub-problems with simpler structure.

In order to give an introduction to the splitting theory, first consider the Cauchy problem for the system of ordinary differential equations with constant coefficients, i.e.

$$\frac{dw}{dt} = (A + B)w(t); \qquad 0 < t \leq T < +\infty;$$

$$w(0) = w_0; \tag{1.50}$$

where $A; B \in \mathbb{R}^{sxs}$ are given matrices,$w_0 \in \mathbb{R}^s$ is a given vector and the unknown function is $w: [0; T) \rightarrow R^s$: Further, we define a usually small number $\tau \ll T$ such that $T = N.\tau.$

Splitting methods can be classified as Classical and Iterative methods.

1.5.15 Classical Methods and Iterative splitting methods

The classical methods are introduced by discussing the sequential splitting methods while the iterative splitting methods are treated by discussing the additive iterative splitting methods.

Lie-Trotter splitting and additive splitting are considered as first order splitting methods, Strang splitting (Strang, 1963) and symmetrically weighted splitting are considered as second order splitting methods.

1.5.16 First Order Splitting: Lie-Trotter Splitting

First order operator splitting method is described, which is called Lie-Trotter splitting. Lie-Trotter splitting is introduced as a method, which solves two sub-problems sequentially on subintervals $[t^n, t^{+1}n]$, where $n = 0,1, \dots, N-1, t^0 = 0$ and $t^N = T$. The different subproblems are connected via the initial conditions.

1.5.17 First Order Splitting: Additive Splitting

This method is based on a simple idea: the different sub-problems are solved by using the same initial function. The study obtained the split solution by the use of these results and the initial condition. Considering the problem (1.50), in the computation of split solutions of the two sub-problems are added, and the initial condition is subtracted from the sum. In this manner obtaining a splitting method where the different sub-problems have no effect on each other. The additive splitting method solves two sub-problems sequentially on sub-intervals. The additive splitting is seen to be a first order method.

1.5.18 Second Order Splitting: Strang Splitting

One of the most popular and widely used operator splitting method is Strang splitting (or Strang-Marchuk operator splitting method) (Marchuk, 1968). By the small modification it is possible to make the splitting algorithm second order accurate. Strang splitting is a numerical method of solving ordinary differential equations (ODEs). It is named after Gilbert Strang. The essential idea is that a complex ODE can be decomposed into multiple simpler ODEs. Each ODE can be advanced independently, and then the total change would be the sum of all individual changes. To demonstrate this idea, suppose an equation of the form (1.51) below, (Trotter, 1959)

$$\frac{dy}{dx} = L_1(y) + L_2(y) \tag{1.51}$$

where L_1, L_2 are differential operators. Suppose further that had we dropped either of the differential operators on the right hand side we would be left with equations that were much simpler to solve.

$$\frac{dy}{dx} = L_{1,2}(y) \tag{1.52}$$

Starting from some point where value of the function is known $y(x_0)$and advance it to the next point $x_0 + \Delta x$ according to the simplified, reduced ODEs, to get

$$y(x_0 + \Delta x) = e^{L_{1,2}(y(x_0))} y(x_0) \tag{1.53}$$

Combining these two advancement operators yields the value of the function at the next point, according to the complete ODE

$$y(x_0 + \Delta x) = e^{L_1(y(x_0))} e^{L_2(y(x_0))} y(x_0) \tag{1.54}$$

and it is seen that Strang splitting gives second order accuracy.

1.5.19 Second Order Splitting: Symmetrically Weighted Splitting

For non-commuting operators, the Lie-Trotter splitting is not symmetric with respect to the operators A and B, and it has first order accuracy. However in many practical cases splitting of higher-order accuracy is required. This can be achieved by the following modified splitting method, called Symmetrically Weighted Splitting which is already symmetrical with respect to the operators. The sequential operator splitting method solves two sub-problems sequentially on sub-intervals

1.5.20 Higher Order Splitting Method

The higher order operator splitting methods are used for more accurate computations, but also with respect to more computational steps. These methods are often performed in quantum dynamics to approximate the evolution operator.

1.5.21 Splitting as a discretization method

The use of any splitting to the problem (1.50) in fact results in a discretization process: while the true solution $w(t)$ of (1.50) is defined on the time interval $[0, T]$, the splitting solution $w_{sp}^{N}(n\tau)$ is defined on the mesh (Geiser J. , 2001)

$\Omega_\tau \emptyset := \{n.\tau, n = 0,1 \dots N\}$. Clearly, the splitting discretization process can be written in the form

$$w^{n+1} = C(\tau)w^n;$$

$$w^0 = w_0 \tag{1.55}$$

where the notation $w^n = w_{SP}^{N}(n\tau)$ has been used. Here C(τ) denotes an operator (matrix) which corresponds to the applied discretization. For the different splitting introduced above, some definitions are given as follows:

1.5.22 Sequential splitting

$$C_{ss}(\tau) = exp(\tau B)\, exp(\tau A); \tag{1.56}$$

1.5.23 Strang splitting (or Strang-Marchuk splitting)

$$C_{st}(\tau) = exp(\tfrac{\tau}{2} A)\, exp(\tau B)\, exp(\tfrac{\tau}{2} A); \tag{1.57}$$

1.5.24 Symmetrically Weighted Sequential Splitting (SWSS)

$$C_{SWSS}(\tau) = \frac{1}{2}[exp(\tau B)\, exp(\tau A) + exp(\tau A)\, exp(\tau B)] \quad (1.58)$$

This means that the local splitting error can be identified with the local (approximation) error. Therefore, it is quite natural to use the following

Definition 1.4 [p-th order accuracy]

A splitting is of p-th order accurate if for the local splitting error the relation

$$\left\|Err_{sp}(\tau)\right\| = O(\tau^{p+1}) \quad (1.59)$$

holds.

In the following the order of accuracy for the different splitting is defined.

- The sequential splitting is of first order accurate.
- The Strang splitting a cumbersome (but simple) calculation has second order accuracy.
- The Symmetrically Weighted Sequential Splitting (SWSS) is of second order accuracy.

In virtue of the general theory of abstract numerical method, the following can be introduced:

Definition 1.5 A splitting process to the ACP problem (1.49) is called consistent if for its solution $w(t)$ and for all fixed $t \in [0;\ T)$ the relation

$$\lim_{\tau\to 0} \left\| \left(\frac{C(\tau)-I}{\tau} - (A + B) \right) w(t) \right\| = 0 \quad (1.60)$$

holds, where the operator (matrix) $C(\tau)$ represents the splitting process.

- The sequential splitting is consistent on the class of the solution of ODE's system.
- The Strang splitting, the SWSS and the WSS are consistent on the class of the solution of ODE's system.

It follows from the abstract theory of numerical methods, the consistency itself doesn't yield the convergence. Therefore the following can be introduced:

Definition 1.6 [Stability]

A splitting with the operator $C(\tau)$ is stable if there exists a constant $K > 0$

such that the relation $\|C(\tau)^n\| \leq K$ (1.61)

holds for all $n \in \mathbb{N}$ such that $n\tau \leq T$.

The necessary condition (Neumann condition) of the stability is

$\rho(C(\tau)) \leq 1 + C_\tau$ (1.62)

for all $\tau \in (0;\ \tau_0)$. For the symmetric matrices $C(\tau)$ the Neumann condition is necessary and sufficient condition of the stability. Taking note that the condition

$\|C(\tau)\| \leq 1 + C_\tau$ (1.63)

is a sufficient condition of the stability. The contractivity of the operators

$exp(\tau A)$ and $exp(\tau B)$, i.e.

$\|exp(\tau A)\| \leq 1;\ \|exp(\tau B)\| \leq 1$ (1.64)

in some norm, is clearly a sufficient condition of the stability (1.61).

Lemma 1.1

The sequential splitting is unconditionally stable for the ODE's system (1.50).

The proof is given by (Istvan, 2003)

Lemma 1.2

The sequential splitting is unconditionally convergent on the class of the solutions of the ODE's system (1.50).

The proof is given by (Istvan, 2003)

1.6 Consistency, Convergence and Stability

1.6.1 Numerical Errors

A numerical error is either of two kinds of error in a calculation. The first (a rounding error) is caused by the finite precision of computations involving floating-point values. Increasing the number of digits allowed in a representation reduces the magnitude of possible round-off errors, but any representation limited to finitely many digits will still cause some degree of round-off error for un-countable real numbers.

The second type of error (sometimes called the truncation error) is the difference between the exact mathematical solution and the approximate solution. Suppose, that we have defined an equidistant mesh {xi} and let us consider first a local error which arises from only one step of some numerical scheme.

A difference

$$\varepsilon_{i+1} = u(x_{i+1}) - u_{i+1} \tag{1.65}$$

is said to be a local discretization error in the point x_{i+1}. Here $u(x_{i+1}))$ is an exact solution of the problem in the point x_{i+1} whereas u_{i+1}describes a value in this point, calculated using the numerical approximation. In other words, the local discretization error can be interpreted as a residuum, if one put the numerical solution into the exact one. Now, if we put the Taylor expansion in the vicinity of the point $(x_i, u(x_i))$ into the equation of interest, one gets the information how fast the local error tends to zero with the spacing Δx.

This observation leads to the definition of the so-called consistency order:

One says, that a numerical scheme possess a consistency order p, if

$$|\varepsilon_{i+1}| \leq C \Delta x^{p+1}, \quad i = 0,1,2,\ldots, \tag{1.66}$$

where C is a constant.

As mentioned above, the local error gives information about the accuracy of the numerial scheme, i.e., about the error in one its step.

At the end of calculation one can calculate an accumulated or a global discretization error in the point x_{i+1}:

$$e_{i+1} = u(x_{i+1}) - u_{i+1} \tag{1.67}$$

The value of the global error gives information about convergence of the approximation to the exact solution of the problem if the spacing value Δx tends to zero. A numerical scheme is said to be convergent, if for the global error e_i one can write

$$max_{i=1\ldots n}|e_i| \to 0 \text{ for } \Delta x \to 0. \tag{1.68}$$

The scheme posseses a convergence order p, if $max_{i=1\ldots n}|e_i| \leq C \Delta x^p$, (1.69)

where C is a constant.

Notice: At a first glance the global error tends to zero with the decreasing of Δx, so the mesh should be refined. However decreasing of spatial distance, Δx , leads to the increase of the rounding off error. Another point to emphasize is that decreasing of Δx can lead to instability of the numerical scheme in question.

1.6.2 Stability

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, for instance, rounding error does not increase in time. Unfortunately, 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 PDE's. The standard method for linear PDE's was proposed by John von Neumann in 1947 and is based on the representation of the rounding error in form of the Fourier series.

1.6.3 Von Neumann stability analysis

Consider the following notation: $u^{j+1} = T[u^j]$. (1.70)

Here T is a nonlinear operator, depending on numerical scheme in question.

The successive application of T results in a consequence of values

$u^{(0)}, u^{(1)}, u^{(2)}, \ldots$,

that approximate the exact solution of the problem. As was mentioned above, at each time step we add a small error $\varepsilon^{(j)}$, i.e.,

$u^{(0)} + \varepsilon^{(0)}, u^{(1)} + \varepsilon^{(1)}, u^{(2)} + \varepsilon^{(2)}, \ldots$,

where $\varepsilon^{(j)}$is a cumulative rounding error at time t_j. Thus we obtain

$u^{(j+1)} + \varepsilon^{(j+1)} = T(u^{(j)} + \varepsilon^{(j)})$ (1.71)

After linearization of the last equation (we suppose that Taylor expansion for T is possible) the linear equation for the perturbation takes the form:

$$\varepsilon^{(j+1)} = \frac{\partial T(u^{(j)})}{\partial u^{(j)}} \varepsilon^{(j)} := G\varepsilon^{(j)} \quad (1.72)$$

This equation is called error propagation law, whereas the linearization matrix G is said to be an amplification matrix. The stability of the numerical scheme depends now on the eigenvalues g_μ of G. In other words, the scheme is stable if and only if

$$|g_\mu| \leq 1 \; \forall \mu$$

The question now is how this information can be used in practice. The first point to emphasize is that in general one deals with the $u(x_i, t_j) := u_{ij}$, so one can write

$$\varepsilon_i^{(j+1)} = \sum_{i'} G_{ii'} \varepsilon_{i'}^{(j)}, \quad (1.73)$$

Where

$$G_{ii'} = \frac{\partial T(u^{(j)})_i}{\partial u_{i'}^{(j)}}.$$

For the values $\varepsilon_i^{(j)}$ (rounding error at the time step tj in the point xi) one can display as a Fourier series:

$$\varepsilon_i^{(j)} = \sum_k e^{Ikx_j} \check{\varepsilon}^{(j)}(k), \quad (1.74)$$

where I depicts the imaginary unit whereas $\check{\varepsilon}^{(j)}(k)$ are the Fourier coefficients. An important point is, that the functions e^{Ikx_j} are Eigen functions of the matrix G, so the last expansion can be interpreted as the expansion in Eigen functions of G. Thus, for the practical point of view one take the error $\varepsilon_i^{(j)}$ just exact as

$$\varepsilon_i^{(j)} = e^{Ikx_j}$$

The substitution of this expression into the Eq. (1.5.9) results in the following relation

$$\varepsilon_i^{(j+1)} = g(k)e^{Ikx_j} = g(k)\varepsilon_i^{(j)} \tag{1.75}$$

Thus e^{Ikx_j} is an eigenvector corresponding to the eigenvalue $g(k)$. The value $g(k)$ is often called an amplification factor. Finally, the stability criteria is given as

$$|g(k)| \leq 1 \quad \forall k \tag{1.76}$$

This criteria is called Von Neumann stability criteria

Theorem 1.2 [Lax equivalence theorem]

States that for a well posed initial value problem, a finite difference approximation to it is convergent if and only if it is consistent and stable.

The proof is given by (Strikwerda, 1989)

1.7 Background information of the problem

Burgers' equation is a fundamental partial differential equation from fluid mechanics. It occurs in various areas of applied mathematics, such as modeling of fluid dynamics and traffic flow. It is named after Johannes Martinus Burgers (1895–1981). It relates to the Navier-Stokes equation for incompressible flow with the pressure term removed (Burgers, 1948)

One of the major challenges in the field of complex systems is a thorough understanding of the phenomenon of turbulence. Direct numerical simulations (DNS) have substantially contributed to our understanding of the disordered flow phenomena inevitably arising at high Reynolds numbers. However, a successful theory of turbulence is still lacking which would allow to predict features of technologically important phenomena like turbulent mixing, turbulent convection, and turbulent combustion on the basis of the fundamental fluid dynamical equations. This is due to the fact that already the evolution equation for the simplest fluids, which are the so-called Newtonian incompressible fluids, have to take

into account nonlinear as well as nonlocal properties.

Nonlinearity stems from the convective term and the pressure term, whereas non-locality enters due to the pressure term.

In 1939 the dutch scientist (Burgers, 1948) simplified the Navier-Stokes equation by just dropping the pressure term. The equation can be investigated in one spatial dimension (Physicists like to denote this as 1+1 dimensional problem in order to stress that there is one spatial and one temporal coordinate):

$$\frac{\partial}{\partial t}u(x,t) + u(x,t)\frac{\partial}{\partial x}\,u(x,t) = \nu\frac{\partial^2}{\partial x^2}u(x,t) + F(x,t) \tag{1.77}$$

Note that usually the Burgers equation is considered without external force $F(x,t)$.

The Burgers equation (1.77) is nonlinear and one expects to find phenomena similar to turbulence. However, as it has been shown by (Hopf, 1950) and (Cole, 1951) the homogeneous Burgers equation lacks the most important property attributed to turbulence: The solutions do not exhibit chaotic features like sensitivity with respect to initial conditions. This can be explicitly shown using the Hopf-Cole transformation which transforms Burgers equation into a linear parabolic equation. From the numerical point of view, however, this is of importance since it allows one to compare numerically obtained solutions of the nonlinear equation with the exact one.

This comparison is important to investigate the quality of the applied numerical schemes. Furthermore, the equation has still interesting applications in physics and astrophysics. Some of the applications include:

- Growth of interfaces: Decomposition models
- Hopf-Cole transformation which maps the solution of the Burgers equation (1.77) into the heat equation

1.8 Problem Statement

The study considers the two dimensional coupled Burgers' equation of the form (1.3). Which is an important equation to Mathematician and Scientist in study of diffusion and convectional flows yet for many years has proved to be time consuming and tedious to solve. Various approaches have been used with different success levels. No direct approach has been used to solve this equation because of its complexity. Hopf-Cole transformation is one of the approaches that have been used with minimal success, also and numerical approaches have been used with varied levels of accuracy. There is very little work has been reported on the use of hybrid finite difference schemes in the context of operation splitting. Moreover the stability, consistency and convergence for such an approach has been studied but with varying results.

1.9 Objectives

1.9.1 General Objective

The study developed hybrid finite difference schemes resulting from operator splitting for solving two dimensional coupled Burgers equation.

1.9.2 Specific Objectives

The study developed the following finite difference schemes:

i) the pure Crank-Nicholson (CN) scheme

ii) the hybrid Crank-Nicholson - Du - Fort and Frankel (CN-DF) scheme

iii) the hybrid Crank-Nicholson – Lax - Friedrichs' (CN-LF) scheme

iv) the hybrid Crank-Nicholson - Lax - Friedrichs'-Du-Fort and Frankel (CN-LF-DF) scheme

These schemes are developed using operator splitting for use to solve the coupled Burgers' system in two dimensions and solution analyzed for accuracy, stability and consistency.

1.10 Significance and Justification of Study

The research carried out the study to determine the solution of a general two dimensional system of Burgers equation. Burgers' equation is a fundamental partial differential equation from fluid mechanics. It occurs in various areas of applied mathematics, such as modeling of fluid dynamics and traffic flow. It relates to the Navier-Stokes equation for incompressible flow with the pressure term removed. Thus this solution is very useful to scientist, engineers and mathematicians who have interest to research on fluid dynamics, traffic flows among other similar flows hence the need to research.

In this study hybrid finite difference scheme with Crank-Nicholson as the "parent" because it is unconditionally stable is used and thus the resulting scheme is stable and also because no work has been done on the use of hybrid finite difference scheme with context of operator splitting to solve the coupled Burgers equation.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

In this chapter, literature review is presented on works done and published on Finite difference, Operator Splitting with a special focus on Burgers equation.

2.2 Finite Difference

Several research has been done on use of finite difference methods to solve partial differential equations, to mention a few (Mitchel & Grffiths, 1980; Ames, 1992; Jain, 2004) describe the three methods: Crank-Nicholson, Lax-Fredichs' and Du-Fort and Frankel methods for finding the numerical solution of partial differential equations. Lax-Fredichs' and Du-Fort and Frankel methods are explicit while the Crank-Nicholson is implicit.

Baruch (1995) proved the stepwise stability for a finite difference scheme for the heat equation with an integral constraint. The resulting matrix is non-symmetric and does not have the usual band structure. The proof is based on the method of matrix analysis. The eigenvalues of several matrices are found explicitly or their location described precisely. This method relies upon the relationship of the characteristic polynomials of these matrices with orthogonal polynomial

Finite difference method is best known method and consists of replacing each derivative by a difference quotient in the classic formulation (Chen, 2013). In his notes he said that 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. According to him the drawback of the finite difference methods is accuracy and flexibility. Standard finite difference methods

requires more regularity of the solution (e.g.$u \in C^2(\Omega)$) and the triangulation (e.g. uniform grids). He said that difficulties also arise in imposing boundary conditions. He gave an overview of the Finite difference formula, Boundary conditions, error estimate and Cell centered finite difference methods.

2.2.1 Finite Difference Formula

Chen (2013) discussed the Poisson equation posed on the unit square $\Omega = (0,1) \times (0,1)$ Variable coefficients and more complex domains were discussed in finite element methods. He explained the popular difference formulas at an interior node x_j for a discrete function u include:

- The backward difference,
- The forward difference,
- The centered difference and
- The centered second difference.

He used the above difference formulation, especially the centered second difference to approximate the Laplace operator at an interior node $(x_i;\ y_j)$.

2.2.2 Boundary Conditions

Chen (2013) discussed how to deal with boundary conditions in finite difference methods. He said that the Dirichlet boundary condition is relatively easy and that the Neumann boundary condition requires the ghost points.

- Dirichlet boundary condition
- Neumann boundary condition

The Poisson equation with the two boundary conditions was clearly illustrated.

2.2.3 Error Estimate

In his explanation he said that in order to analyze the error, we need to put the problem into a norm space.

2.2.4 Cell Centered Finite Difference Methods

In this section, he considered (with clear illustrations) Finite Difference methods for the Poisson equation at cell centers.

2.3 Du Fort-Frankel

Du Fort-Frankel can be traced back to 1953 when it was presented as one of the numerical methods for solving the heat equation with periodic boundary conditions, (Du Fort & Frankel, 1953). The scheme is explicit, and it is unconditionally stable for the initial value problem. The generalized Du Fort-Frankel scheme has been tested for the Burger's equation using the 4^{th} order accurate operators and the scheme developed was run with $\Delta x = \Delta t = 0.1$, and the error was found to be 16 times smaller in accordance with the 4^{th} order accuracy steady state solution and was found with no sign of instability (Gottlieb & Gustafsson, 1976). It however faces consistency problems for large values of Δt (Mitchel & Grffiths , 1980)

2.4 Crank-Nicholson Method

In his paper Teukolsky (1999) explained that the iterated Crank-Nicholson method has become a popular algorithm in numerical relativity. He showed that one should carry out exactly two iterations and no more. While the limit of an infinite number of iterations is the standard Crank-Nicholson method, and that it can be worse to do more than two iterations, and it never helps.

According to Tadjeran (2007) the Crank–Nicholson method is a widely used method to obtain numerical approximations to the diffusion equation due to its accuracy and unconditional stability. He explained that when the diffusion coefficient is not a constant, the general approach is to obtain a discretization for the PDE in the same manner as the case for constant coefficients. He showed that the manner of this discretization may impact the stability of the resulting method and could lead to instability of the numerical solution and that the classical Crank–Nicholson method will fail to be unconditionally stable if the diffusion coefficient is computed at the time grid-points instead of at the midpoints of the temporal subinterval. In their research a numerical example was presented and compared with the exact analytical solution to examine its divergence. In conclusion they explained that to get unconditional stability for the Crank–Nicholson approximation, the diffusion coefficients should be computed at the midpoint of the each temporal subinterval in the integration process. That this will provide an unconditionally stable difference method with a local truncation error that is $O[(\Delta t)^2] + O[(\Delta x)^2]$, and will therefore be second-order convergent. Otherwise, under certain conditions the Crank–Nicholson method will be unstable if the diffusion coefficients is computed at the two endpoints of each time subintervals during integration process.

In their paper, (Ashyralyev *et al.*, 2010) considered the mixed problem for one-dimensional parabolic equation with non-smooth data generated by the blood flow through glycocalyx on the endothelial cells. Stable numerical method was developed and solved by using the r-modified Crank–Nicholson schemes. Numerical analysis was given for a constructed problem. Differential equations were discretized and an algorithm was established to solve the differential equations using MATLAB software. For parabolic equations with non-smooth initial data, they constructed the modified Crank–Nicholson

difference schemes which are convergent. The convergence and smoothness property of modified Crank–Nicholson difference schemes were investigated. The application of this numerical method was applied to a real problem with numerical data taken from the hospital. The velocities inside the core flow region and porous flow region was calculated. Then, the wall shear stress values and the drag forces on the glycocalyx structure was calculated. The results was analysed to understand the effect of shear stresses and drag force on the mechano-transduction, which is defined as the transduction of mechanical forces to the endothelial cells which caused biochemical signalling inside the cells.

2.5 Burgers equation

The Burgers' equation was first introduced by Bateman (Bateman, 1915), who derived the steady state solution for the one-dimensional equation and was studied in details by Burgers (Burgers, 1948). Linearized parabolic equations appear as models in heat flow and gas dynamics. Finite difference solutions of these equations are found by using ordinary discretization, (Mitchel & Grffiths, 1980; Ames, 1992).

In the paper of (Weinan, 1992), a general framework is presented for analyzing numerical methods for the evolutionary equations that admit semi-group formulations. This framework was then applied to spectral and Pseudo-spectral methods for the Burgers' equation, using trigonometric, Chebyshev, and Legendre polynomials. Optimal order of convergence was obtained, which implies the spectral accuracy of these methods. In conclusion, they remarked that the approach presented applies to much more general situations such as the Burgers' equation with Neumann boundary condition. However, it is essential that the semi-group be regularizing (here we required that the semi-group be analytic). Therefore this approach does not apply to hyperbolic equations. In particular,

the estimates presented in their paper deteriorate when the viscosity coefficient goes to zero. In the limit of zero viscosity, discontinuities form spontaneously in the solution, even with smooth initial data. Numerical experience indicates that the methods analyzed in the paper do not converge for discontinuous solutions, unless some kind of smoothing is done.

Certain types of boundary value problems can be solved by replacing the differential equation by the corresponding finite difference equation and then solving the latter by a process of iteration. These methods have been used by many mathematicians according to Jain (2004).

The paper of (Han *et al.*, 2006) discussed the numerical solution of Burgers' equation on unbounded domains. Two artificial boundaries were introduced and boundary conditions were obtained on the artificial boundaries, which are in nonlinear forms. Then the original problem was reduced to an equivalent problem on a bounded domain. Finite difference method was applied to the reduced problem, and some numerical examples are given to show the effectiveness of the new approach. The artificial boundary method was been applied to the Burgers' equation on unbounded domains. Using the Cole-Hopf transformation they obtained the boundary conditions on the artificial boundaries. The boundary conditions are in nonlinear forms. With the artificial boundaries, they solved the original unbounded problem in a much smaller domain, the computational work was be greatly reduced. The numerical examples showed that the new approach is very effective, the numerical solutions converge fast to the exact solutions.

Analytic solution of the Burgers' equation involves series solutions which converge very slowly for small values of viscosity constant according to Idris (2007).

The article of (Hairer & Voss, 2010) was devoted to the numerical study of various finite difference approximations to the stochastic Burgers equation. Their particular interest was in the one-dimensional case is the situation where the driving noise is white both in space and in time. They demonstrated that in this case, different finite difference schemes converge to different limiting processes as the mesh size tends to zero. A theoretical explanation of this phenomenon was given and they formulated a number of conjectures for more general classes of equations, supported by numerical evidence. In the article studies on several finite difference schemes for the viscous stochastic Burgers equation were made.

In the paper of Jiang and Wang (2010), an improved numerical solution of the Burgers' equation is presented based on the cubic B-spline quasi-interpolation and the compact finite difference method. At first the cubic B-spline quasi-interpolation and the compact finite difference method are introduced. Moreover, the numerical scheme was presented, by using the derivative of the quasi-interpolation to approximate the spatial derivative and a two-order compact scheme to approximate the time derivative. The accuracy of their scheme was demonstrated by two problems. The advantage of the resulting scheme was simple with better accuracy, so it was easy to implement. The improved solution could be significant to the applications such as modeling of fluid dynamics and traffic flow. In the paper, they gave an efficient numerical scheme for Burgers' equation which is a very important fluid dynamic model. The numerical scheme was presented based on the cubic B-spline quasi-interpolation and the compact finite difference method. From the test

examples their scheme is feasible and the accuracy is better than other quasi-interpolation methods. The implementation of the method is easy and acceptable.

In their paper Rao and Yadav (2010), they expressed the solution of an initial value problem for a nonhomogeneous Burgers equation in terms of a family of self-similar solutions of the heat equation. In their work they allowed bounded and compactly supported initial profiles.

Coupled non-linear Burgers' equations in two dimensions is a special form of incompressible Navier-Stokes equations without the pressure term and the continuity equation (Vineet *et al.*, 2011).

In the past several years, numerical solutions to one-dimensional Burgers' equations have attracted a lot of attention of the researchers (Srivastava *et al.*, 2011). Other researchers have solved the coupled two dimensional Burgers' equation and is mentioned in (Ali, 2009; Basto *et al.*, 2009; Rashidi & Erfani, 2009; Zheng, 2010) among others.

In their paper Kweyu et *al*. (2012) they generated varied sets of exact initial and Dirichlet boundary conditions for the 2-D Burgers' equations from general analytical solutions via Hopf-Cole transformation and separation of variables. These conditions were then used for the numerical solutions of this equation using finite difference methods (FDMs) and in particular the Crank-Nicolson (CN) and the explicit schemes.

In their paper Kutluay and Yagmurlu (2012) the modified bi-quintic B-spline base functions were proposed and successfully applied to the two-dimensional unsteady Burgers' equation using the Galerkin method to obtain its numerical solutions. The accuracy of the numerical scheme is examined by the error norms L_2 and L_0. The obtained

numerical results were compared with the exact ones. The obtained results were found to be in good agreement with the exact ones.

In their work Srivastava et *al.* (2013) proposed a numerical approach for solving one dimensional coupled nonlinear Burgers' equations using a fully implicit finite-difference scheme. The efficiency and accuracy of the scheme was demonstrated taking three test examples. The numerical results showed that fully implicit finite-difference scheme performs well in the case of 1D coupled Burgers' equation. The proposed method proved to be highly accurate as compared to the other numerical methods and showed excellent agreement with the exact solution.

In the paper of Iltaf et *al.* (2013) they proposed a mesh-free technique for the numerical solution of the two dimensional Burger's equation. Collocation method using the Radial Basis Functions (RBFs) was coupled with first order accurate finite difference approximation. Different types of RBFs were used for this purpose. Performance of the proposed method was successfully tested in terms of various error norms. In the case of non-availability of exact solution, performance of the new method is compared with the results obtained from the existing methods available in the literature. The elementary stability analysis was established theoretically and is also supported by numerical results. In conclusion they said that the approach is generally more applicable than the traditional methods like finite difference and finite element methods as it can be scaled up to higher dimensions in convenient way and does not require nodal or elements connectivity. The problems presented in the paper suggested that mesh-free approximation methods should be considered as one of possible ways of solving nonlinear partial differential equations similar to Burgers' equations.

Consideration of Burger's equation as a fundamental partial differential equation from fluid mechanics was done by (Shafiqul et *al.*, 2014). They showed derivation of Navier-Stokes equation, Burger's equation and numerical methods of Burger's equation. They also showed that numerical result based on the explicit central difference scheme agrees with basic qualitative behavior of viscuss Burger's equation.

In the paper of Aminikhah and Moradia (2014) a numerical method for solving the systems of variable-coefficient coupled Burgers' equation was used. The method was based on two-dimensional Legendre wavelets. Two-dimensional operational matrices of integration were introduced and then employed to find a solution to the systems of variable-coefficient coupled Burgers' equation. Two examples were presented to illustrate the capability of the method. It was shown that the numerical results were in good agreement with the exact solutions for each problem. The aim of the paper was to develop two-dimensional Legendre wavelets for obtaining the solutions of systems of variable-coefficient coupled Burgers' equation. The illustrative examples included demonstrate that we have achieved a method is a very effective and useful technique for finding approximate solutions of these systems. The method is fully described possible error and analyzed. The two-dimensional operational matrices of integration were used to find the solution of the system of variable-coefficient coupled Burgers' equation. In the method, the problem under study reduced to a system of linear or nonlinear algebraic equations. The two examples presented illustrated the capability and simplicity of the method and the close comparison of the obtained results with those of the exact solutions showed that their method is a highly promising method for various classes of both linear and nonlinear

systems of partial differential equations. Here, the computations associated with these examples were performed by the package Maple 13.

The paper of Amruta and Vikas (2014) presents a new analytical method called Variational Homotopy Perturbation Method (VHPM), which is a combination of the well-known Variational Iteration method (VIM) and the Homotopy Perturbation method (HPM) for solving the one-dimensional Burger's equation. Two test problems were presented to demonstrate the efficiency and the accuracy of the proposed method. The numerical solutions obtained were compared with the exact solution. They proved that the method did not require spatial discretization or restrictive assumptions and was proved to be free from round-off errors and therefore reduced the numerical computation significantly. The results reveal that the Variational Homotopy Perturbation Method is very effective and convenient to solve non-linear partial differential equations. According to them, a comparison was made to show that method has small size of computation in comparison with the computational size required in other numerical methods and its rapid convergence shows that the method is reliable and introduces a significant improvement in solving partial differential equation.

2.6 Operator splitting

Operator splitting is a powerful method for numerical investigation of complex models. It involves splitting complex problem into a sequence of simpler tasks that can be called split sub-problems (Yesim, 2010). Espen in his thesis (Espen, 2011) discussed numerical quadratures in one and two dimensions, which was followed by a discussion regarding the differentiation of general operators in Banach spaces. In the research he investigated the Godunov and Strang method numerically for the viscous Burgers' equation and the

Korteweg-de Vries (KdV) equation and presented different numerical methods for the sub-equations from the splitting. They discovered that the Operator splitting methods work well numerically for the two equations. Also in his thesis, Yesim (Yesim, 2010) studied consistency and stability of the operator splitting methods. He concentrated on how to improve the classical operator splitting methods via Zassenhaus product formula.

In their research paper (Qimiao & Onyx , 1998) discussed and presented a new efficient numerical method for three-dimensional hydrodynamic computations. The method is based on the operator splitting method and combined with Eulerian–Lagrangian method, finite element method and finite difference method. To increase the efficiency and stability of the numerical solutions, the operator splitting method was employed to partition the momentum equations into three parts, according to physical phenomena. A time step was divided into three time sub-steps. In the first sub-step, advection and Coriolis force were solved using the explicit Eulerian–Lagrangian method. In the second sub-step, horizontal diffusion was approximated by implicit Finite Element Method (FEM) in each horizontal layer. In the last sub-step, the continuity equation was solved by implicit FEM, and vertical diffusion and pressure gradient were discretized by implicit Finite Difference Method (FDM) in each nodal column. The stability analysis showed that the method is unconditionally stable. A number of numerical experiments were performed. The results simulated by the scheme developed agree well with analytical solutions and the other documented model results. They concluded that the method is efficient for 3D shallow water flow computations and fully fits complicated configurations.

In his presentation Delgado (2013) explained with illustration that operator splitting is used because they are fast, flexible and non-intrusive as compared to fully coupled operators which are slow, inflexible and intrusive. He discussed the flavors of operator splitting which includes: Classical methods, Alternating Direction Implicit (ADI) method and Iterative splitting methods.

Classical methods involve coupling through initial conditions and it includes:

- Lie-Trotter Sequential Splitting and
- Strang-Marchuk Splitting.

The ADI methods involve coupling through source terms and it includes:

- Peaceman-Rachford and
- Douglas-Rachford methods

The ADI methods can be classified as:

- Monotone Operators (which is unconditionally stable),
- Commutative Operators and
- Non-commutative operators.

The advantages of the ADI includes:

- Reduced storage requirements and
- Tri-diagonal Solvers

While the disadvantages includes:

- Non-trivial extension to more than 2 operators,

- May not converge for non-monotone operators and
- Cannot get better than $O(\Delta t^2)$ accuracy

The Iterative Splitting methods involve the use of Fixed Point Iteration and the accuracy is limited only by time integration method and has no additional "splitting" error. It has the advantages that includes:

- Arbitrary accuracy is achievable,
- Even for non-commutative operators and
- Extendible to multiple operators

The Disadvantages include:

- Multiple iterations per time-step and
- Only converges for contraction mappings

Hockbruck and Osterman (2005) demonstrated and discussed time integration due to operator splitting for linear 1-D parabolic equations. Le Veque and Oliger (1983) describes additive operator splitting for hyperbolic partial difference equations.

(Ames, 1992; Mitchel & Grffiths, 1980) describes additive operator splitting for parabolic equation which are more than one dimensional. Another splitting method mentioned by Mitchel and Grffiths (1980) is called second order and was developed by Strang in the 1960s. Istvan (2003) gave an elaborate discussion of operator splitting for parabolic equations. Splitting method has been used by (Evje & Hvistendahl , 1999) to find the numerical solution of convection –diffusion equation.

In his lecture notes (Boyd et *al.*, 2013) explained clearly the Operator Splitting Methods.

The main idea is to write F as $F = A + B$, with A and B maximal monotone called

operator splitting and to solve using methods that require evaluation of resolvents

$R_A = (I + \lambda A)^{-1}, \quad R_B = (I + \lambda B)^{-1}$

He explained the Peaceman-Rachford splitting as un-damped iteration

$z^{k+1} = C_A C_B (z^k)$

that doesn't converge in general case; need C_A or C_B to be contraction.

Nodal Operator Splitting Adaptive Finite Element Algorithms for the Navier-Stokes Equations has been explained in the paper of (Wille, 1998). Solution algorithms for solving the Navier–Stokes equations without storing equation matrices were developed. The algorithms operate on a nodal basis, where the finite element information is stored as the co-ordinates of the nodes and the nodes in each element. Temporary storage is needed, such as the search vectors, correction vectors and right hand side vectors in the conjugate gradient algorithms which are limited to one-dimensional vectors. The nodal solution algorithms consist of splitting the Navier–Stokes equations into equation systems which he solved sequentially. In the pressure split algorithm, the velocities were found from the diffusion–convection equation and the pressure is computed from these velocities. The computed velocities are then corrected with the pressure gradient. In the velocity–pressure split algorithm, a velocity approximation was first found from the diffusion equation. This velocity was corrected by solving the convection equation. The pressure was then found from these velocities. Finally, the velocities are corrected by the pressure gradient. The nodal algorithms are compared by solving the original Navier–Stokes equations. The pressure split and velocity–pressure split equation systems were solved using ILU preconditioned conjugate gradient methods where the equation matrices were stored, and by using diagonal preconditioned conjugate gradient methods without storing the equation matrices.

The advantage of explicit time marching schemes for solving partial differential equations is that the prediction of the future solution from earlier solutions is done without having to solve a system of equations. Therefore, computer memory for storing the equation matrices is not required and it is possible to solve larger problems with significantly more degrees of freedom.

Explicit time marching schemes are extensively used for hyperbolic equation systems, while implicit time schemes are most frequently used for solving parabolic equation systems. Implicit algorithms offer a higher degree of stability, which is required when diffusion is included in the flow equations. However, implicit algorithms usually require more work and more storage, as an equation system has to be solved.

The paper of (Luo, 1996) describes a finite element implementation of an operator-splitting algorithm for solving transient/steady turbulent flows and presents solutions for the turbulent flow in an axisymmetric $180°$ narrowing bend, a benchmark problem dealt with at the 1994 World User Association in Applied Computational Fluid Dynamics (WUA-CFD) annual meeting. Three $k-\varepsilon$ based models were used: the standard linear $k-\varepsilon$ model, a non-linear $k-\varepsilon$ model and an RNG $k-\varepsilon$ model. Flow separation after the bend, as observed in the experiment, was predicted by the RNG model and by both the linear and non-linear $k-\varepsilon$ models with van Driest mixing length wall functions. Good agreement with experimental data of pressure distribution on bending walls was obtained by the present numerical simulation. Results showed that there is very little difference between the linear and non-linear $k-\varepsilon$ models in terms of predicted velocity fields and that the non-linearities mainly affected the distribution of turbulent normal stress and pressure, in analogy to the effect of second-order viscoelastic fluid models on laminar

flow. Both the linear and non-linear $k-\varepsilon$ models fail to predict any flow separation if logarithmic wall functions are used. Calculations showed that the operator splitting time-stepping algorithm is very efficient, robust and stable. The usual problem of negative values of k and ε can be either completely avoided or minimized in the semi-implicit time-stepping scheme by limiting the magnitude of time steps. The RNG model predicted flow separation without any difficulty; the linear and non-linear $k-\varepsilon$ models failed to predict separation with a logarithmic profile one-layer wall function but succeeded with the van Driest mixing length wall function. This gives a clear explanation of why some reports at World User Association in Applied Computational Fluid Dynamics (WUA-CFD) 1994 concluded that standard $k-\varepsilon$ does not predict separation while others at the same meeting reached the opposite conclusion. Different groups at WUA-CFD 1994 used different wall functions, but the present work has demonstrated that, with everything else the same, different wall functions can make the difference between complete failure and success in predicting flow separation for the WUA-CFD benchmark problem. There is very little difference between the linear and non-linear k-& models in terms of predicted velocity fields and the difference in predicted distributions of turbulent normal stress and pressure is much more pronounced, in analogy to the effect of second-order viscoelastic fluid models on laminar flow. The comparison of predictions with experimental data on pressure distribution on solid walls showed good overall agreement, except in the high-pressure-gradient (normal to the wall) region near the detachment point on the inside wall. He discovered that the large discrepancy between all predictions and experiment in the high-pressure-gradient near-wall region highlights the failure of using universal wall functions in non-universal flow regions and the need for better wall treatment in general.

In his work Wille (1998) explicit and implicit time marching schemes were combined to solve the Navier–Stokes equations. The conjugate gradient method was used to solve the parabolic diffusion equation and the Laplace equation was used for the pressure implicitly. The forward Euler method was used to solve the convection equation. The investigation was an initial demonstration of the ability of the operator splitting algorithms.

Iterative operator splitting method was used by (Nurcan & Gamze, 2011) to solve numerically the mathematical model for capillary formation in tumor angiogenesis problem. The method was based on first splitting the complex problem into simpler sub-problems. Then each sub-equation was combined with iterative schemes. The algorithms were obtained by applying the proposed method to the given model problem. The explicit local error bounds were derived to show consistency. In their paper they also explained the stability by constructing the stability functions. The obtained numerical results showed that iterative splitting method provides high accuracy and efficiency with respect to other classical methods in the literature. He concluded that the iterative splitting method is superior to the others, because of the following reasons:

- It includes all operators in each sub-equation unlike the traditional operator splitting methods. This is physically the best and hence they obtain the consistent approximations after each inner step.
- It reduces the local splitting error by using more iteration steps to obtain higher order accuracy.
- It has a small constant in the local splitting error with respect to the method of lines.

- It gives a better performance at long time with respect to the finite difference method.

As a result, this application showed that the iterative operator splitting method gives high convergence and small error and it is quite easy to apply to the model problem. The consistency and stability analysis are also studied easily.

Computing solutions of convection-diffusion equations, especially in the convection dominated case, is an important and challenging problem that requires development of fast, reliable numerical methods (Chertock, Kurganov & Petrova, 2004). In their report they used a second-order fast explicit operator splitting (FEOS) method based on the Strang splitting. The main idea of the method was to solve the parabolic problem via a discretization of the formula for the exact solution of the heat equation, which was realized using a conservative and accurate quadrature formula. The hyperbolic problem was solved by a second-order finite-volume Godunov-type scheme. The fast explicit operator splitting (FEOS) method was applied to the one- and two-dimensional systems modelling two phase multicomponent flow in porous media. Their results demonstrated that the method achieves a remarkable resolution and accuracy in a very efficient manner, that is, when only few splitting steps are performed.

In the work of (Krishnan et *al.*, 2006), noise removal in digital images was investigated. The importance of this problem lies in the fact that removal of noise is a necessary pre-processing step for other image processing tasks such as edge detection, image segmentation, image compression, classification problems, image registration etc. A number of different approaches have been proposed in the literature. In this particula work, a non-linear PDE-based algorithm consisting of two steps: flow field smoothing of the normal vectors, followed by image reconstruction. They used a finite-difference based

additive operator-splitting method that allows for much larger time-steps. This resulted in an efficient method for noise-removal that was shown to have good visual results. The energy is studied as an objective measure of the algorithm performance.

In their report (Blanes et *al.*, 2006) presented a family of symplectic splitting methods especially tailored to solve numerically the time-dependent Schrodinger equation. When discretised in time, this equation can be recast in the form of a classical Hamiltonian system with a Hamiltonian function corresponding to a generalized high-dimensional separable harmonic oscillator. The structure of the system allowed them to build highly efficient symplectic integrators at any order. The methods were found to be accurate, easy to implement and very stable in comparison with other standard symplectic integrators. In conclusion, they claimed that the processing technique leads to extraordinarily efficient symplectic split operator methods for the Schrodinger equation.

In the article of (Geiser & Noack, 2008) they considered iterative operator-splitting methods for non-linear differential equations with respect to their eigenvalues. The main feature of their idea was the fixed-point iterative scheme that linearizes their underlying equations. Based on the approximated eigenvalues of such linearized systems they chose the order of the operators for their iterative splitting scheme. The convergence properties of such mixed method were studied and demonstrated. They confirmed with numerical applications the effectiveness of their scheme in comparison with the standard operator-splitting methods by providing improved results and convergence rates. They then applied their results to deposition processes. In conclusion they presented a new method to solve complicated mixed coupled partial differential equations. Based on a standard method they derived different new methods and reorder the operators for different scales. Such a reordering reduced the decomposition error. The more hyperbolic behaviour of the

equations leads to an increment of the iteration steps of their method. At least they obtained a second order method. Such iterative splitting method can balance the different behaviour of the underlying operators. They explained that the one operator smoothen the solution process, while the other operator decreases the smoothness. Further they said that the balance between the implicit and explicit discretization with the iterative splitting method is a new method that overcomes to the mixed behaviour in an un-split method.

A new analytical approach to operator splitting for equations of the type

$u_t = Au + uu_x$

where A is a linear differential operator such that the equation is well-posed was provided by Holden et *al.* (2011). Particular examples include the viscous Burgers' equation, the Korteweg–de Vries (KdV) equation, the Benney–Lin equation, and the Kawahara equation. They showed that the Strang splitting method converges with the expected rate if the initial data are sufficiently regular. In particular, for the Korteweg–de Vries (KdV) equation they obtained second-order convergence in H^r for initial data in H^{r+5} with arbitrary $r \geq 1$.

Presentation of an accurate numerical method for a large class of scalar, strongly degenerate convection–diffusion equations was presented in the work of (Holden et *al.*, 2000). Important subclasses discussed are hyperbolic conservation laws, porous medium type equations, two-phase reservoir flow equations, and strongly degenerate equations coming from the recent theory of sedimentation–consolidation processes. The method is based on splitting the convective and the diffusive terms. The nonlinear, convective part was solved using front tracking and dimensional splitting, while the nonlinear diffusion part is solved by an implicit–explicit finite difference scheme. In addition, one version of the implemented operator splitting method has a mechanism built in for detecting and

correcting unphysical entropy loss, which may occur when the time step is large. They explained that the mechanism helps to gain large time step ability for practical computations. A detailed convergence analysis of the operator splitting method was given they presented numerical experiments with the method of modelling secondary oil recovery and sedimentation–consolidation processes. They demonstrated that the splitting method resolves sharp gradients accurately, may use large time steps, has first order convergence, exhibits small grid orientation effect, has small mass balance errors and is efficient. The paper demonstrated the applicability of operator splitting methods to a large class of scalar, convection-dominated problems. Due to explicit tracking of shocks in the hyperbolic part, and the possibility of correcting unphysical entropy loss, the methods give very accurate representation of sharp gradient phenomena. This was demonstrated for both reservoir simulation and simulation of sedimentation processes. The main ingredient in the splitting methods is the use of a large-step front-tracking method, which is very computationally efficient compared with standard finite difference methods. In the parabolic steps, they mostly used a simple explicit finite difference method.

In the paper of (Borah et *al.*, 2012) they studied mathematical models for fluid flow which often involve systems of convection-diffusion equations as a main ingredient. They explained that in operator splitting - one splits the time evolution into partial steps to separate the effects of convection and diffusion. In their paper they showed that the temporal splitting error can be significant when there is a shock present in the solution, this is well-understood for scalar convection – diffusion equation. The paper deals the motivation for operator splitting methods is that it is easy to combine efficient methods for solving the convection step with efficient methods for the diffusive step. In the case for convection dominated systems, they explained that it is a major advantage to be able

to use an accurate and efficient hyperbolic solver developed for tracking discontinuous solutions. Furthermore, combining this with efficient methods for the diffusive-step, they obtained a powerful and efficient numerical method which is well suited for solving parabolic problems with sharp gradients. They gave the following conclusions:

- They demonstrated numerically that operator splitting methods for systems of convection-diffusion equations in one-space dimensions.
- It has a tendency to be too diffusive near viscous shock waves.
- The scalar corrected operator splitting method is to use wave structure from the convection step to identify where the nonlinear splitting error (or entropy loss) occurs.
- The potential error is compensated for in the diffusion step (or in a separate correction step).
- In case of scalar case, the splitting error is closely related to the local linearization introduced implicitly in the convection steps due to the use of an entropy condition.

The numerical results demonstrate that the corrected operator splitting method is significantly more accurate than the corresponding operator splitting method when the splitting step is large and the solution consists of (moving) viscous shock waves.

The work done by Yesim (Yesim & Gamze, 2015) provided an error analysis of the operator splitting method of the Lie-Trotter type applied to the Burgers-Huxley equation of the form:

$u_t + \alpha u u_x - \varepsilon u u_{xx} = \beta (1-u)(u-\gamma)u.$

They showed that the Lie-Trotter splitting method converges with the expected rate in $H^s(\mathbb{R})$, where $H^s(\mathbb{R})$ is the Sobolev space and s is an arbitrary non-negative integer. They split the equation into linear and nonlinear parts and applied numerical methods for the sub-problems. They then present errors and confirmed the theoretical results with the numerical example.

2.7 Hybrid Finite Difference method

Hybrid Schemes with Crank-Nicholson was first introduced by 2009 to solve the 1-D heat equation using operator splitting by modifying it (Koross et *al.*, 2009). In their paper they developed hybrid finite difference method resulting from operator splitting for solving the modified form and proved that there is an improvement in efficacy of the Crank-Nicholson scheme when the Lax-Friedrich's and Du Fort and Frankel discretization's are used on it. They concluded in their research findings that the Crank-Nicholson-Lax-Friedrich-Du For and Frankel is the most accurate method for solving 1-D heat equation.

In their paper Kweyu *et al.* (2014) they developed a hybrid Crank-Nicolson and Du Fort and Frankel scheme. The hybrid Crank-Nicolson and Du Fort and Frankel scheme was developed by introducing the Du Fort and Frankel properties into the Crank-Nicolson scheme. The scheme is three-level and is also unconditionally stable. The Numerical solutions from the hybrid scheme were obtained by the use of MATLAB software. By use of L1 error, they determined that the hybrid scheme is fifth order accurate in space and produces better results in comparison to the pure Crank-Nicolson and the pure Du Fort and Frankel schemes.

2.8 Alternating Direction Implicit Method

Alternating Direction Implicit Formulation of the Differential Quadrature Method (ADI-DQM) has been used in the past to solve the Burgers equation in two-dimension. The numerical results showed that the ADI-DQM has the higher accuracy and convergence as well as the less computation workload by using few grid points (Al-Saif et *al.*, 2012)

Peaceman and Rachford (1955) explained that in mathematics, the alternating direction implicit (ADI) method is a finite difference method for solving parabolic and elliptic partial differential equations. It is mostly used to solve the problem of heat conduction or solving the diffusion equation in two or more dimensions. The traditional method for solving the heat conduction equation is the Crank–Nicholson method. This method can be quite costly. The advantage of the ADI method is that the equations that have to be solved in every iteration have a simpler structure and are thus easier to solve. In the method linear diffusion equation in three dimensions is considered. The implicit Crank–Nicholson method produces a stable finite difference method.

There are more refined ADI methods such as the methods of (Douglas, 1962), or the f-factor method of (Chang, 1991) which can be used for three or more dimensions.

2.9 Stability, Consistence and Convergence

Several authors including: (Rao, 2005; Rahman, 1998; Chapra & Canale, 1998) discussed the stability for both the explicit and implicit schemes. While (Mitchel & Grffiths , 1980; Ames, 1992; Morton & Mayers, 2005) discussed the Neumann method and the matrix method. These authors have established that the explicit method due to Schmidt given in equation (1.42) is stable for $0 < r \leq \frac{1}{2}$ and $\lambda = 1$.

The explicit method due to Du-Fort and Frankel and all the implicit methods are unconditionally stable. Ames (1992) quotes Flat who proved that stability in uniform sense does not hold for $r = \frac{k}{h^2} > R$

where R depends upon the length L of the bar in a heat conduction problem. For this reason treatment of convergence of the convergence of the Crank-Nicholson equation, one must use a procedure based upon a combination of Duhamel's principle and harmonic analysis.

In the research solutions of the two dimensional coupled Burgers equations using finite difference and operator splitting techniques, which has not been done following the survey above, are found.

CHAPTER THREE

METHODOLOGY

3.1 Introduction

In this chapter we explain step by step approach used to solve our problem in this research.

The research has used MATLAB application to generate the results of the hybrid methods formed and compared results with the proposed solution developed by (Kweyu et *al.*, 2012).

Operator splitting technique on two dimensional coupled Burgers' equation (1.3) with boundary and initial conditions (1.4) and (1.5) respectively and taking $\mathrm{D} = \{(\mathrm{x}, y) | 0 \leq x \leq 1, 0 \leq y \leq 1\}$ has been employed.

The operator splitting method is one of the most powerful methods to solve the abstract Cauchy problems. The main idea is to lead the complex problem to the sequence of sub-problems with simpler structure. For our case each part of the coupled Burgers' equation is split into simpler units which can be easily discretized.

In mathematics, discretization concerns the process of transferring continuous functions, models and equations into discrete counterparts. This process is usually carried out as a first step toward making them suitable for numerical evaluation and implementation on digital computers.

Finite-difference methods (FDM) are numerical methods for solving differential equations by approximating them with difference equations, in which finite differences approximate the derivatives. FDMs are thus discretization methods. Today, FDMs are the dominant approach to numerical solutions of partial differential equations.

From the resulting split operator form, we develop hybrid finite difference methods whose "parentage" is the Crank-Nicholson method starting with the Pure Crank-Nicholson (CN) then Hybrid Crank-Nicholson - Lax-Frediechs'(CN-LF), Hybrid Crank-Nicholson Du-Fort and Frankel (CN-DF) and finally Hybrid Crank-Nicholson - Lax-Frediechs' - Du-Fort and Frankel schemes(CN-LF-DF). The Hybrid schemes are developed by attaching a part from another finite difference method into the developed pure CN scheme, for instance CN-LF is the pure CN with a part from LF attached to form the hybrid CN-LF. The properties of Crank-Nicholson are thus improved by the introduction of the Lax-Fredrich's properties, thus the name "hybrid".

3.2 Outline of the Operator Splitting Technique for a Parabolic Equation

We outline the operator splitting technique for the equation of the form:

$$u_t = Lu, \ (\mathrm{a} \le \mathrm{x} \le \mathrm{b}) \times (\mathrm{c} \le \mathrm{y} \le \mathrm{d}) \times (0 \le \mathrm{t} \le \mathrm{T}) \tag{3.1}$$

$$u(x, y, 0) = u_0(x, y) \tag{3.2}$$

where $u = u(x, y, t)$ and L is a sum of differential operators and a times with source functions.

Consider the Taylor's expansion

$$\begin{aligned} u(x, y, t + k) &= u(x, y, t) + k\frac{\partial}{\partial t}u(x, y, t) + \frac{k^2}{2!}\frac{\partial^2}{\partial t^2}u(x, y, t) + \cdots \\ &= \left(1 + k\frac{\partial}{\partial t} + \frac{k^2}{2!}\frac{\partial^2}{\partial t^2} + \cdots\right) u(x, y, t) \\ &= e^{k\frac{\partial}{\partial t}}\left(u(x, y, t)\right) \end{aligned} \tag{3.3}$$

In equation (3.3) we can replace $\frac{\partial}{\partial \mathrm{t}}$ by L that is

$$u(x, y, t + k) = e^{kL}\, u(x, y, t) \tag{3.4}$$

The exact solution of the equation (3.1) - (3.2) at the grid point ($x = mh, y = lq, t = nk$) is $u(x, y, t)$ with h, q and k being the grid spacing in the x- direction, y- direction and t- direction respectively. m, l and k are intergers. $m = l = n = 0$ is the origin. The approximate solution at this point is denoted by $U_{m,l,n}$. We can then write the finite difference (FD) approximation of equation (3.3) as:

$$U_{m,l,n+1} = e^{kL} U_{m,l,n} \tag{3.5}$$

In equations (3.3) and (3.5) e^{kL} is called the solution operator for equation (3.1) L is replaced by finite difference approximation. In equation (3.4) L can be taken to be a sum of differential operators with respect to x.

If $L = L_1 + L_2 + L_3 + \cdots + L_S = \sum_{i=1}^{S} L_i$

Then equation (3.5) can be written as

$$U_{m,l,n+1} = e^{k \sum_{i=1}^{S} L_i} U_{m,l,n}$$

$$= e^{k\,(L_1 + L_2 + L_3 + \cdots + L_S)} U_{m,l,n} \tag{3.6}$$

$$= (e^{kL_1})(e^{kL_2}) \ldots . (e^{kL_{s-1}})(e^{kL_s}) U_{m,l,n} \tag{3.7}$$

$$= \prod_{i=1}^{S} e^{kL_i} U_{m,l,n} \tag{3.8}$$

The approximate solution can be obtained from equation (3.7) by first solving

$$U_{m,l,n+1}^{(s)} = e^{kL_s} U_{m,l,n} \tag{3.9}$$

and then using this solution we can find

$$U_{m,l,n+1}^{(s-1)} = e^{kL_{s-1}} U_{m,l,n} \tag{3.10}$$

We go on like this until we attain

$$U_{m,l,n+1}^{(1)} \tag{3.11}$$

which is actually the approximate solution of equation (3.1)

Considering the 2-D Burgers equation of the form (1.3)

Here s=4

And so

$L = L_1 + L_2 + L_3 + L_4$

Let

$L_1 = -\frac{1}{h} u_{m,l,n} \delta_x$

$L_2 = -\frac{1}{q} v_{m,l,n} \delta_y$

$L_3 = \frac{1}{Reh^2} {\delta_x}^2$

$L_4 = \frac{1}{Req^2} {\delta_y}^2$

From equation (3.7) the approximate solution is found by

$$U_{m,l,n+1} = (e^{kL_1})\ (e^{kL_2})(e^{kL_3})(e^{kL_4})\ U_{m,l,n} \tag{3.12}$$

$$U_{m,l,n+1} = \left(1+kL_1+\tfrac{1}{2}k^2{L^2}_1+..\right)\left(1+kL_2+\tfrac{1}{2}k^2{L^2}_2+..\right)\left(1+kL_3+\tfrac{1}{2}k^2{L^2}_3+..\right)\left(1+kL_4+\tfrac{1}{2}k^2{L^2}_4+..\right)$$

$$\approx 1 + kL_1 + kL_2 + kL_3 + kL_4 + L_1L_2k^2 + L_1L_3k^2 + L_1L_4k^2 + L_2L_3k^2 + L_2L_4k^2 +$$

$$L_3L_4k^2 + \frac{1}{2}k{L_1}^2 + \frac{1}{2}k{L_2}^2 + \frac{1}{2}k{L_3}^2 + \frac{1}{2}k{L_4}^2 \tag{3.13}$$

The scheme developed is used to solve the coupled Burgers' equations.

3.3 Test of Stability, Consistency and convergence

The developed scheme is analyzed by a study of the error. The stability, consistency and convergence is studied. Von Newman condition is used to test for stability.

3.3.1 Consistency

One says, that a numerical scheme possess a consistency order p, if

$$|\varepsilon_{i+1}| \leq C\ \Delta\ x^{p+1}, \quad i = 0,1,2,\ldots, \tag{3.14}$$

where C is a constant.

The value of the global error gives information about convergence of the approximation to the exact solution of the problem if the spacing value $\Delta\, x$ tends to zero.

3.3.2 Convergence

A numerical scheme is said to be convergent, if for the global error e_i one can write

$$max_{i=1\ldots n}|e_i| \to 0 \text{ for } \Delta\, x \to 0. \tag{3.15}$$

The scheme posseses a convergence order p, if

$$max_{i=1\ldots n}|e_i| \leq C\ \Delta\, x^p, \tag{3.16}$$

where C is a constant.

3.3.3 Stability

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, for instance, rounding error does not increase in time. The standard method for linear PDE's was proposed by John von Neumann in 1947 and is based on the representation of the rounding error in form of the Fourier series (Gurevich, 2008).

3.3.4 Von Neumann stability analysis

Consider the following notation:

$$u^{j+1} = T[u^j]. \tag{3.17}$$

Here T is a nonlinear operator, depending on numerical scheme in question.

The successive application of T results in a consequence of values

$u^{(0)}, u^{(1)}, u^{(2)}, ..$

that approximate the exact solution of the problem. As was mentioned above, at each time step we add a small error $\varepsilon^{(j)}$, i.e.,

$u^{(0)} + \varepsilon^{(0)}, u^{(1)} + \varepsilon^{(1)}, u^{(2)} + \varepsilon^{(2)}, ..$

where $\varepsilon^{(j)}$is a cumulative rounding error at time t_j. Thus we obtain

$$u^{(j+1)} + \varepsilon^{(j+1)} = T(u^{(j)} + \varepsilon^{(j)}) \tag{3.18}$$

After linearization of the last equation (we suppose that Taylor expansion for T is possible) the linear equation for the perturbation takes the form:

$$\varepsilon^{(j+1)} = \frac{\partial T(u^{(j)})}{\partial u^{(j)}} \varepsilon^{(j)} = G\varepsilon^{(j)} \tag{3.19}$$

This equation is called error propagation law, whereas the linearization matrix G is said to be an amplification matrix. The stability of the numerical scheme depends now on the eigenvalues g_μ of G.

The scheme is stable if and only if

$$\left|g_\mu\right| \leq 1 \; \forall \mu \tag{3.20}$$

which is called the Von Neumann Stability Criteria.

CHAPTER FOUR

RESULTS AND DISCUSSIONS

4.1 Introduction

In this chapter the schemes are developed and the solutions of the schemes are presented and discussed.

4.2 Pure Crank-Nicholson (CN) Scheme

First it is necessary that the pure Crank-Nicholson method resulting from this splitting is developed. This is because other hybrid methods are derived from it. Thus the Crank-Nicholson method is as follows:

$$L_1 U_{m,l,n} = -\frac{1}{2h} U_{m,l,n} \delta_x (U_{m,l,n} + U_{m,l,n+1})$$

$$= -\frac{1}{2h} U_{m,l,n}\left(U_{m+1,l,n} - 2U_{m,l,n} + U_{m-1,l,n} + U_{m+1,l,n+1} - 2U_{m,l,n+1} + U_{m-1,l,n+1}\right), \quad (4.1)$$

$$L_2 V_{m,l,n} = -\frac{1}{2q} V_{m,l,n} \delta_y (U_{m,l,n} + U_{m,l,n+1})$$

$$= -\frac{1}{2q} V_{m,l,n}\left(U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n} + U_{m,l+1,n+1} - 2U_{m,l,n+1} + U_{m,l-1,n+1}\right), \quad (4.2)$$

$$L_3 U_{m,l,n} = \frac{1}{4Reh^2} \delta_x^2 (U_{m,l,n} + U_{m,l,n+1})$$

$$= \frac{1}{4Reh^2} \delta_x\left(U_{m+1,l,n} - 2U_{m,l,n} + U_{m-1,l,n} + U_{m+1,l,n+1} - 2U_{m,l,n+1} + U_{m-1,l,n+1}\right),$$

$$= \frac{1}{4Reh^2}(U_{m+2,l,n} - 4U_{m+1,l,n} + 6U_{m,l,n} - 4U_{m-1,l,n} + U_{m-2,l,n} + U_{m+2,l,n+1} - 4U_{m+1,l,n+1} + 6U_{m,l,n+1} - 4U_{m-1,l,n+1} \quad (4.3)$$

$$L_4 U_{m,l,n} = \frac{1}{4Req^2} \delta_y^2 (U_{m,l,n} + U_{m,l,n+1})$$

$$= \frac{1}{4Req^2}\delta_y\left(U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n} + U_{m,l+1,n+1} - 2U_{m,l,n+1} + U_{m,l-1,n+1}\right)$$

$$= \frac{1}{4Req^2}(U_{m,l+2,n} - 4U_{m,l+1,n} + 6U_{m,l,n} - 4U_{m,l-1,n} + U_{m,l-2,n} + U_{m,l+2,n+1} -$$

$$4U_{m,l+1,n+1} + 6U_{m,l,n+1} - 4U_{m,l-1,n+1} + U_{m,l-2,n+1} \quad (4.4)$$

$$L_1L_2V_{m,l,n} = \frac{1}{4hq}U_{m,l,n}V_{m,l,n}\,\delta_x\,(\delta_y(U_{m,l,n} + U_{m,l,n+1}))$$

$$= \frac{1}{4hq}U_{m,l,n}V_{m,l,n}\,\delta_x\,(U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n} + U_{m,l+1,n+1} - 2U_{m,l,n+1} +$$

$$U_{m,l-1,n+1})$$

$$= \frac{1}{4hq}U_{m,l,n}V_{m,l,n}\,(U_{m+1,l+1,n} - 2U_{m,l+1,n} + U_{m-1,l+1,n} - 2U_{m+1,l,n} + 4U_{m,l,n} -$$

$$2U_{m-1,l,n} + U_{m+1,l-1,n} - 2U_{m,l-1,n} + U_{m-1,l-1,n} + U_{m+1,l+1,n+1} - 2U_{m,l+1,n+1} +$$

$$U_{m-1,l+1,n+1} - 2U_{m+1,l,n+1} + 4U_{m,l,n+1} - 2U_{m-1,l,n+1} + U_{m+1,l-1,n+1} - 2U_{m,l-1,n+1} +$$

$$U_{m-1,l-1,n+1}) \quad (4.5)$$

$$L_1L_3U_{m,l,n} = -\frac{1}{8h^3Re}U_{m,l,n}\,\delta_x^3\,(U_{m,l,n} + U_{m,l,n+1})$$

$$= -\frac{1}{8h^3Re}U_{m,l,n}\,\delta_x\,(\delta_x\left(\delta_x(U_{m,l,n} + U_{m,l,n+1})\right))$$

$$= -\frac{1}{8h^3Re}U_{m,l,n}\,\delta_x\,\left(\delta_x\left(U_{m+1,l,n} - 2U_{m,l,n} + U_{m-1,l,n} + U_{m+1,l,n+1} - 2U_{m,l,n+1} +\right.\right.$$

$$\left.\left.U_{m-1,l,n+1}\right)\right)$$

$$= -\frac{1}{8h^3Re}U_{m,l,n}\,\delta_x\,\left(U_{m+2,l,n} - 4U_{m+1,l,n} + 6U_{m,l,n} - 4U_{m-1,l,n} + U_{m-2,l,n} +\right.$$

$$\left.U_{m+2,l,n+1} - 4U_{m+1,l,n+1} + 6U_{m,l,n+1} - 4U_{m-1,l,n+1} + U_{m-2,l,n+1}\right)$$

$$= -\frac{1}{8h^3Re}U_{m,l,n}\,\left(U_{m+3,l,n} - 6U_{m+2,l,n} + 14U_{m+1,l,n} - 20U_{m,l,n} + 14U_{m-1,l,n} -\right.$$

$$6U_{m-2,l,n} + U_{m-3,l,n} + U_{m+3,l,n+1} - 6U_{m+2,l,n+1} + 15U_{m+1,l,n+1} - 20U_{m,l,n+1} +$$

$$\left.15U_{m-1,l,n+1} - 6U_{m-2,l,n+1} + U_{m-3,l,n+1}\right) \quad (4.6)$$

$$L_1L_4U_{m,l,n} = -\frac{1}{8hq^2Re}U_{m,l,n}\,\delta_x\,(\delta_y\left(\delta_y(U_{m,l,n} + U_{m,l,n+1})\right))$$

$$= -\frac{1}{8hq^2Re} U_{m,l,n}\, \delta_x\, (\delta_y\big(U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n} + U_{m,l+1,n+1} - 2U_{m,l,n+1} +$$

$$U_{m,l-1,n+1}\big))$$

$$= -\frac{1}{8hq^2Re} U_{m,l,n}\ \delta_x\big(U_{m,l+2,n} - 4U_{m,l+1,n} + 6U_{m,l,n} - 4U_{m,l-1,n} + U_{m,l-2,n} +$$

$$U_{m,l+2,n+1} - 4U_{m,l+1,n+1} + 6U_{m,l,n+1} - 4U_{m,l-1,n+1} + U_{m,l-2,n+1}\big)$$

$$= -\frac{1}{8hq^2Re} U_{m,l,n}\, \big(U_{m+1,l+2,n} - 2U_{m,l+2,n} + U_{m-1,l+2,n} - 4U_{m+1,l+1,n} + 8U_{m,l+1,n} -$$

$$4U_{m-1,l+1,n} + 6U_{m+1,l,n} - 12U_{m,l,n} + 6U_{m-1,l,n} - 4U_{m+1,l-1,n} + 8U_{m,l-1,n} -$$

$$4U_{m-1,l-1,n} + U_{m+1,l-2,n} - 2U_{m,l-2,n} + U_{m-1,l-2,n} + U_{m+1,l+2,n+1} - 2U_{m,l+2,n+1} +$$

$$U_{m-1,l+2,n+1} - 4U_{m+1,l+1,n+1} + 8U_{m,l+1,n+1} - 4U_{m-1,l+1,n+1} + 6U_{m+1,l,n+1} -$$

$$12U_{m,l,n+1} + 6U_{m-1,l,n+1} - 4U_{m+1,l-1,n+1} + 8U_{m,l-1,n+1} - 4U_{m-1,l-1,n+1} +$$

$$U_{m+1,l-2,n+1} - 2U_{m,l-2,n+1} + U_{m-1,l-2,n+1}\big)$$

(4.7)

$$L_2L_3U_{m,l,n} = -\frac{1}{8qh^2Re} V_{m,l,n}\, \delta_y(\delta_x^2(U_{m,l,n} + U_{m,l,n+1})\,)$$

$$= -\frac{1}{8qh^2Re} V_{m,l,n}\, \delta_y\, (U_{m+2,l,n} - 4U_{m+1,l,n} + 6U_{m,l,n} - 4U_{m-1,l,n} + U_{m-2,l,n} +$$

$$U_{m+2,l,n+1} - 4U_{m+1,l,n+1} + 6U_{m,l,n+1} - 4U_{m-1,l,n+1} + U_{m-2,l,n+1})$$

$$= -\frac{1}{8qh^2Re} V_{m,l,n}\, (U_{m+2,l+1,n} - 2U_{m+2,l,n} + U_{m+2,l-1,n} - 4U_{m+1,l+1,n} + 8U_{m+1,l,n} -$$

$$4U_{m+1,l-1,n} + 6U_{m,l+1,n} - 12U_{m,l,n} + 6U_{m,l-1,n} - 4U_{m-1,l+1,n} + 8U_{m-1,l,n} -$$

$$4U_{m-1,l-1,n} + U_{m-2,l+1,n} - 2U_{m-2,l,n} + U_{m-2,l-1,n} + U_{m+2,l+1,n+1} - 2U_{m+2,l,n+1} +$$

$$U_{m+2,l-1,n+1} - 4U_{m+1,l+1,n+1} + 8U_{m+1,l,n+1} - 4U_{m+1,l-1,n+1} + 6U_{m,l+1,n+1} -$$

$$12U_{m,l,n+1} + 6U_{m,l-1,n+1} - 4U_{m-1,l+1,n+1} + 8U_{m-1,l,n+1} - 4U_{m-1,l-1,n+1} +$$

$$U_{m-2,l+1,n+1} - 2U_{m-2,l,n+1} + U_{m-2,l-1,n+1})$$

(4.8)

$$L_2L_4V_{m,l,n} = -\frac{1}{8q^3Re} V_{m,l,n}\delta_y^3(U_{m,l,n} + U_{m,l,n+1})$$

$$= -\frac{1}{8q^3Re}V_{m,l,n}\delta_y(\delta_y(U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n} + U_{m,l+1,n+1} - 2U_{m,l,n+1} + U_{m,l-1,n+1}))$$

$$= -\frac{1}{8q^3Re}V_{m,l,n}\delta_y(U_{m,l+2,n} - 4U_{m,l+1,n} + 6U_{m,l,n} - 4U_{m,l-1,n} + U_{m,l-2,n} + U_{m,l+2,n+1} - 4U_{m,l+1,n+1} + 6U_{m,l,n+1} - 4U_{m,l-1,n+1} + U_{m,l-2,n+1})$$

$$= -\frac{1}{8q^3Re}V_{m,l,n}(U_{m,l+3,n} - 6U_{m,l+2,n} + 15U_{m,l+1,n} - 20U_{m,l,n} + 15U_{m,l-1,n} - 6U_{m,l-2,n} + U_{m,l-3,n} + U_{m,l+3,n+1} - 6U_{m,l+2,n+1} + 15U_{m,l+1,n+1} - 20U_{m,l,n+1} + 15U_{m,l-1,n+1} - 6U_{m,l-2,n+1} + U_{m,l-3,n+1})$$

(4.9)

$$L_3L_4V_{m,l,n} = -\frac{1}{8q^3Re}V_{m,l,n}\delta_x^2\delta_y^2(U_{m,l,n} + U_{m,l,n+1})$$

$$= -\frac{1}{8q^3Re}V_{m,l,n}\delta_y^2(U_{m+2,l,n} - 4U_{m+1,l,n} + 6U_{m,l,n} - 4U_{m-1,l,n} + U_{m-2,l,n} + U_{m+2,l,n+1} - 4U_{m+1,l,n+1} + 6U_{m,l,n+1} - 4U_{m-1,l,n+1} + U_{m-2,l,n+1})$$

$$= -\frac{1}{8q^3Re}V_{m,l,n}\delta_y(U_{m+2,l+1,n} - 2U_{m+2,l,n} + U_{m+2,l-1,n} - 4U_{m+1,l+1,n} + 8U_{m+1,l,n} - 4U_{m+1,l-1,n} + 6U_{m,l+1,n} - 12U_{m,l,n} + 6U_{m,l-1,n} - 4U_{m-1,l+1,n} + 8U_{m-1,l,n} - 4U_{m-1,l-1,n} + U_{m-2,l+1,n} - 2U_{m-2,l,n} + U_{m-2,l-1,n} + U_{m+2,l+1,n+1} - 2U_{m+2,l,n+1} + U_{m+2,l-1,n+1} - 4U_{m+1,l+1,n+1} + 8U_{m+1,l,n+1} - 4U_{m+1,l-1,n+1} + 6U_{m,l+1,n+1} - 12U_{m,l,n+1} + 6U_{m,l-1,n+1} - 4U_{m-1,l+1,n+1} + 8U_{m-1,l,n+1} - 4U_{m-1,l-1,n+1} + U_{m-2,l+1,n+1} - 2U_{m-2,l,n+1} + U_{m-2,l-1,n+1})$$

$$= -\frac{1}{8q^3Re}V_{m,l,n}(U_{m+2,l+2,n} - 4U_{m+2,l+1,n} + 6U_{m+2,l,n} - 4U_{m+2,l+1,n} + U_{m+2,l-2,n} - 4U_{m+1,l+2,n} + 16U_{m+1,l+1,n} - 24U_{m+1,l,n} + 16U_{m+1,l-1,n} - 4U_{m+1,l-2,n} + 6U_{m,l+2,n} - 24U_{m,l+1,n} + 36U_{m,l,n} + 6U_{m,l-2,n} - 4U_{m-1,l+2,n} + 16U_{m-1,l+1,n} - 24U_{m-1,l,n} + 16U_{m-1,l-1,n} - 4U_{m-1,l-2,n} + U_{m-2,l+2,n} - 4U_{m-2,l+1,n} + 6U_{m-2,l,n} - 4U_{m-2,l-1,n} + U_{m-2,l-2,n} + U_{m+2,l+2,n+1} - 4U_{m+2,l+1,n+1} + 6U_{m+2,l,n+1} - 4U_{m+2,l+1,n+1} + U_{m+2,l-2,n+1} - 4U_{m+1,l+2,n+1} + 16U_{m+1,l+1,n+1} - 24U_{m+1,l,n+1} + 16U_{m+1,l-1,n+1} - 4U_{m+1,l-2,n+1} + 6U_{m,l+2,n+1} - 24U_{m,l+1,n+1} + 36U_{m,l,n+1} + 6U_{m,l-2,n+1} - 4U_{m-1,l+2,n+1} + 16U_{m-1,l+1,n+1} - 24U_{m-1,l,n+1} + 16U_{m-1,l-1,n+1} - 4U_{m-1,l-2,n+1} + U_{m-2,l+2,n+1} - 4U_{m-2,l+1,n+1} + 6U_{m-2,l,n+1} - 4U_{m-2,l-1,n+1} + U_{m-2,l-2,n+1})$$

(4.10)

$$L_1{}^2\, U_{m,l,n} \;=\; \frac{1}{4h^2}\, U_{m,l,n}^2 \delta_x^2 \left(U_{m,l,n} \;+\; U_{m,l,n+1}\right)$$

$$= \frac{1}{4h^2}\, U_{m,l,n}^2 \left(U_{m+2,l,n} - 4U_{m+1,l,n} + 6U_{m,l,n} - 4U_{m-1,l,n} \;+ U_{m-2,l,n} + U_{m+2,l,n+1} \;-\right.$$

$$\left. 4U_{m+1,l,n+1} + 6U_{m,l,n+1} - 4U_{m-1,l,n+1} + U_{m-2,l,n+1}\right) \qquad (4.11)$$

$$L_2{}^2\, V_{m,l,n} \;=\; \frac{1}{4q^2}\, V_{m,l,n}^2 \delta_y^2 (U_{m,l,n} \;+\; U_{m,l,n+1}\,)$$

$$= \frac{1}{4q^2}\, V_{m,l,n}^2 (U_{m,l+2,n} - 4U_{m,l+1,n} + 6U_{m,l,n} - 4U_{m,l-1,n} \;+ U_{m,l-2,n} +$$

$$U_{m,l+2,n+1} \;- 4U_{m,l+1,n+1} + 6U_{m,l,n+1} - 4U_{m,l-1,n+1} \;+ U_{m,l-2,n+1}\,) \qquad (4.12)$$

$$L_3{}^2\, U_{m,l,n} \;=\; \frac{1}{16Re^2h^4}\, \delta_x^4 \left(U_{m,l,n} \;+\; U_{m,l,n+1}\right)$$

$$= \frac{1}{16Re^2h^4}\, \delta_x^2 \left(U_{m+2,l,n} - 4U_{m+1,l,n} + 6U_{m,l,n} - 4U_{m-1,l,n} \;+ U_{m-2,l,n} + U_{m+2,l,n+1} \;-\right.$$

$$\left. 4U_{m+1,l,n+1} + 6U_{m,l,n+1} - 4U_{m-1,l,n+1} + U_{m-2,l,n+1}\right)$$

$$= \frac{1}{16Re^2h^4}\, \delta_x \left(U_{m+3,l,n} - 6U_{m+2,l,n} + 15U_{m+1,l,n} - 20U_{m,l,n} + 15U_{m-1,l,n} - 6U_{m-2,l,n} +\right.$$

$$\left. U_{m-3,l,n} + U_{m+3,l,n+1} - 6U_{m+2,l,n+1} + 15U_{m+1,l,n+1} - 20U_{m,l,n+1} + 15U_{m-1,l,n+1} -\right.$$

$$\left. 6U_{m-2,l,n+1} + U_{m-3,l,n+1}\right)$$

$$= \frac{1}{16Re^2h^4} \left(U_{m+4,l,n} - 8U_{m+3,l,n} + 28U_{m+2,l,n} - 56U_{m+1,l,n} + 70U_{m,l,n} - 56U_{m-1,l,n} +\right.$$

$$\left. 28U_{m-2,l,n} - 8U_{m-3,l,n} + U_{m-4,l,n} + U_{m+4,l,n+1} - 8U_{m+3,l,n+1} + 28U_{m+2,l,n+1} -\right.$$

$$\left. 56U_{m+1,l,n+1} + 70U_{m,l,n+1} - 56U_{m-1,l,n+1} + 28U_{m-2,l,n+1} - 8U_{m-3,l,n+1} + U_{m-4,l,n+1}\right) \qquad (4.13)$$

$$L_4{}^2\, U_{m,l,n} \;=\; \frac{1}{16Re^2q^4}\, \delta_y^4 \left(U_{m,l,n} \;+\; U_{m,l,n+1}\right)$$

$$= \frac{1}{16Re^2q^4}\, \delta_y^2 \left(U_{m,l+2,n} - 4U_{m,l+1,n} + 6U_{m,l,n} - 4U_{m,l-1,n} \;+ U_{m,l-2,n} + U_{m,l+2,n+1} \;-\right.$$

$$\left. 4U_{m,l+1,n+1} + 6U_{m,l,n+1} - 4U_{m,l-1,n+1} \;+ U_{m,l-2,n+1}\right)$$

$$= \frac{1}{16Re^2q^4}\delta_y\big(U_{m,l+3,n} - 6U_{m,l+2,n} + 15U_{m,l+1,n} - 20U_{m,l,n} + 15U_{m,l-1,n} -$$

$$6U_{m,l-2,n} + U_{m,l-3,n} + U_{m,l+3,n+1} - 6U_{m,l+2,n+1} + 15U_{m,l+1,n+1} - 20U_{m,l,n+1} +$$

$$15U_{m,l-1,n+1} - 6U_{m,l-2,n+1} + U_{m,l-3,n+1}\big)$$

$$= \frac{1}{16Re^2q^4}\big(U_{m,l+4,n} - 8U_{m,l+3,n} + 28U_{m,l+2,n} - 56U_{m,l+1,n} + 70U_{m,l,n} - 56U_{m,l-1,n} +$$

$$28U_{m,l-2,n} - 8U_{m,l-3,n} + U_{m,l-4,n} + U_{m,l+4,n+1} - 8U_{m,l+3,n+1} + 28U_{m,l+2,n+1} -$$

$$56U_{m,l+1,n+1} + 70U_{m,l,n+1} - 56U_{m,l-1,n+1} + 28U_{m,l-2,n+1} - 8U_{m,l-3,n+1} + U_{m,l-4,n+1}\big)$$

(4.14)

Using equations (4.1)-(4.14) in equation (3.2.15) and letting

$q = h$

$$-\frac{1}{2h} = -\frac{1}{2q} = a$$

$$\frac{1}{4Reh^2} = \frac{1}{4Req^2} = b$$

we have

$$U_{m,l,n+1} = 1 + kaU_{m,l,n}\big(U_{m+1,l,n} - 2U_{m,l,n} + U_{m-1,l,n} + U_{m+1,l,n+1} - 2U_{m,l,n+1} +$$

$$U_{m-1,l,n+1}\big) + kaV_{m,l,n}\big(U_{m,l+1,n} - 2U_{m,l,n} + U_{m,l-1,n} + U_{m,l+1,n+1} - 2U_{m,l,n+1} +$$

$$U_{m,l-1,n+1}\big) + kb\big(U_{m+2,l,n} - 4U_{m+1,l,n} + 6U_{m,l,n} - 4U_{m-1,l,n} + U_{m-2,l,n} + U_{m+2,l,n+1} -$$

$$4U_{m+1,l,n+1} + 6U_{m,l,n+1} - 4U_{m-1,l,n+1} + U_{m-2,l,n+1}\big) + kb\big(U_{m,l+2,n} - 4U_{m,l+1,n} +$$

$$6U_{m,l,n} - 4U_{m,l-1,n} + U_{m,l-2,n} + U_{m,l+2,n+1} - 4U_{m,l+1,n+1} + 6U_{m,l,n+1} -$$

$$4U_{m,l-1,n+1} + U_{m,l-2,n+1}\big) + k^2a^2U_{m,l,n}V_{m,l,n}\big(U_{m+1,l+1,n} - 2U_{m,l+1,n} + U_{m-1,l+1,n} -$$

$$2U_{m+1,l,n} + 4U_{m,l,n} - 2U_{m-1,l,n} + U_{m+1,l-1,n} - 2U_{m,l-1,n} + U_{m-1,l-1,n} +$$

$$U_{m+1,l+1,n+1} - 2U_{m,l+1,n+1} + U_{m-1,l+1,n+1} - 2U_{m+1,l,n+1} + 4U_{m,l,n+1} - 2U_{m-1,l,n+1} +$$

$$U_{m+1,l-1,n+1} - 2U_{m,l-1,n+1} + U_{m-1,l-1,n+1}) + k^2abU_{m,l,n}\ (U_{m+3,l,n} - 6U_{m+2,l,n} +$$

$$15U_{m+1,l,n} - 20U_{m,l,n} + 15U_{m-1,l,n} - 6U_{m-2,l,n} + U_{m-3,l,n} + U_{m+3,l,n+1} -$$

$$6U_{m+2,l,n+1} + 15U_{m+1,l,n+1} - 20U_{m,l,n+1} + 15U_{m-1,l,n+1} - 6U_{m-2,l,n+1} +$$

$$U_{m-3,l,n+1}) + k^2abU_{m,l,n}\,(U_{m+1,l+2,n} - 2U_{m,l+2,n} + U_{m-1,l+2,n} - 4U_{m+1,l+1,n} +$$

$$8U_{m,l+1,n} - 4U_{m-1,l+1,n} + 6U_{m+1,l,n} - 12U_{m,l,n} + 6U_{m-1,l,n} - 4U_{m+1,l-1,n} +$$

$$8U_{m,l-1,n} - 4U_{m-1,l-1,n}\ + U_{m+1,l-2,n} - 2U_{m,l-2,n} + U_{m-1,l-2,n} + U_{m+1,l+2,n+1} -$$

$$2U_{m,l+2,n+1} + U_{m-1,l+2,n+1} - 4U_{m+1,l+1,n+1} + 8U_{m,l+1,n+1} - 4U_{m-1,l+1,n+1} +$$

$$6U_{m+1,l,n+1} - 12U_{m,l,n+1} + 6U_{m-1,l,n+1} - 4U_{m+1,l-1,n+1} + 8U_{m,l-1,n+1} -$$

$$4U_{m-1,l-1,n+1}\ + U_{m+1,l-2,n+1} - 2U_{m,l-2,n+1} + U_{m-1,l-2,n+1}) +$$

$$k^2abV_{m,l,n}\,(U_{m+2,l+1,n} - 2U_{m+2,l,n} + U_{m+2,l-1,n} - 4U_{m+1,l+1,n} + 8U_{m+1,l,n} -$$

$$4U_{m+1,l-1,n} + 6U_{m,l+1,n} - 12U_{m,l,n} + 6U_{m,l-1,n} - 4U_{m-1,l+1,n} + 8U_{m-1,l,n} -$$

$$4U_{m-1,l-1,n} + U_{m-2,l+1,n} - 2U_{m-2,l,n} + U_{m-2,l-1,n} + U_{m+2,l+1,n+1} - 2U_{m+2,l,n+1} +$$

$$U_{m+2,l-1,n+1} - 4U_{m+1,l+1,n+1} + 8U_{m+1,l,n+1} - 4U_{m+1,l-1,n+1} + 6U_{m,l+1,n+1} -$$

$$12U_{m,l,n+1} + 6U_{m,l-1,n+1} - 4U_{m-1,l+1,n+1} + 8U_{m-1,l,n+1} - 4U_{m-1,l-1,n+1} +$$

$$U_{m-2,l+1,n+1} - 2U_{m-2,l,n+1} + U_{m-2,l-1,n+1}) + k^2abV_{m,l,n}(U_{m,l+3,n} - 6U_{m,l+2,n} +$$

$$15U_{m,l+1,n}\ - 20U_{m,l,n} + 15U_{m,l-1,n} - 6U_{m,l-2,n}\ + U_{m,l-3,n} + U_{m,l+3,n+1} -$$

$$6U_{m,l+2,n+1} + 15U_{m,l+1,n+1}\ - 20U_{m,l,n+1} + 15U_{m,l-1,n+1} - 6U_{m,l-2,n+1}\ +$$

$$U_{m,l-3,n+1})\ + k^2ab(U_{m+2,l+2,n} - 4U_{m+2,l+1,n} + 6U_{m+2,l,n} - 4U_{m+2,l+1,n} +$$

$$U_{m+2,l-2,n} - 4U_{m+1,l+2,n} + 16U_{m+1,l+1,n} - 24U_{m+1,l,n} + 16U_{m+1,l-1,n} - 4U_{m+1,l-2,n} +$$

$$6U_{m,l+2,n} - 24U_{m,l+1,n} + 36U_{m,l,n} + 6U_{m,l-2,n} - 4U_{m-1,l+2,n} + 16U_{m-1,l+1,n} -$$

$$24U_{m-1,l,n} + 16U_{m-1,l-1,n} - 4U_{m-1,l-2,n} + U_{m-2,l+2,n} - 4U_{m-2,l+1,n} + 6U_{m-2,l,n} -$$

$$4U_{m-2,l-1,n} + U_{m-2,l-2,n} + U_{m+2,l+2,n+1} - 4U_{m+2,l+1,n+1} + 6U_{m+2,l,n+1} -$$

$$4U_{m+2,l+1,n+1} + U_{m+2,l-2,n+1} - 4U_{m+1,l+2,n+1} + 16U_{m+1,l+1,n+1} - 24U_{m+1,l,n+1} + 16U_{m+1,l-1,n+1} - 4U_{m+1,l-2,n+1} + 6U_{m,l+2,n+1} - 24U_{m,l+1,n+1} + 36U_{m,l,n+1} + 6U_{m,l-2,n+1} - 4U_{m-1,l+2,n+1} + 16U_{m-1,l+1,n+1} - 24U_{m-1,l,n+1} + 16U_{m-1,l-1,n+1} - 4U_{m-1,l-2,n+1} + U_{m-2,l+2,n+1} - 4U_{m-2,l+1,n+1} + 6U_{m-2,l,n+1} - 4U_{m-2,l-1,n+1} + U_{m-2,l-2,n+1}) + \frac{1}{2}ka^2U_{m,l,n}^2(U_{m+2,l,n} - 4U_{m+1,l,n} + 6U_{m,l,n} - 4U_{m-1,l,n} + U_{m-2,l,n} + U_{m+2,l,n+1} - 4U_{m+1,l,n+1} + 6U_{m,l,n+1} - 4U_{m-1,l,n+1} + U_{m-2,l,n+1}) + \frac{1}{2}kb^2\,V_{m,l,n}^2(U_{m,l+2,n} - 4U_{m,l+1,n} + 6U_{m,l,n} - 4U_{m,l-1,n} + U_{m,l-2,n} + U_{m,l+2,n+1} - 4U_{m,l+1,n+1} + 6U_{m,l,n+1} - 4U_{m,l-1,n+1} + U_{m,l-2,n+1}) + \frac{1}{2}kb^2(U_{m+4,l,n} - 8U_{m+3,l,n} + 28U_{m+2,l,n} - 56U_{m+1,l,n} - 56U_{m-1,l,n} + 28U_{m-2,l,n} - 8U_{m-3,l,n} + U_{m-4,l,n} + U_{m+4,l,n+1} - 8U_{m+3,l,n+1} + 28U_{m+2,l,n+1} - 56U_{m+1,l,n+1} - 56U_{m-1,l,n+1} + 28U_{m-2,l,n+1} - 8U_{m-3,l,n+1} + U_{m-4,l,n+1} + 140U_{m,l,n} + 140U_{m,l,n+1} + U_{m,l+4,n} - 8U_{m,l+3,n} + 28U_{m,l+2,n} - 56U_{m,l+1,n} - 56U_{m,l-1,n} + 28U_{m,l-2,n} - 8U_{m,l-3,n} + U_{m,l-4,n} + U_{m,l+4,n+1} - 8U_{m,l+3,n+1} + 28U_{m,l+2,n+1} - 56U_{m,l+1,n+1} - 56U_{m,l-1,n+1} + 28U_{m,l-2,n+1} - 8U_{m,l-3,n+1} + U_{m,l-4,n+1})$$

(4.15)

We shall now discuss the approximation at the boundaries.

The proposed solution of equation (1.3) at any point (x, y, t) according to work by (Kweyu *et al*., 2012) is:

$$u(x,y,t) = \frac{-2y-2\pi e^{-\frac{2\pi^2 t}{Re}}((cos\pi x - sin\,\pi x)sin\,\pi y)}{(Re(100+xy+e^{-\frac{2\pi^2 t}{Re}}((cos\pi x - sin\,\pi x)sin\,\pi y))} \quad (4.16)$$

$$v(x,y,t) = \frac{-2x-2\pi e^{-\frac{2\pi^2 t}{Re}}((cos\pi x - sin\,\pi x)sin\,\pi y)}{(Re(100+xy+e^{-\frac{2\pi^2 t}{Re}}((cos\pi x - sin\,\pi x)sin\,\pi y))} \quad (4.17)$$

and so

$$u_x = \frac{2\pi^2 e^{-\frac{2\pi^2 t}{Re}}\left((sin\pi x+cos\,\pi x)\sin\pi y\right)\left(Re\left(100+xy+e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x-sin\,\pi x)sin\,\pi y\right)\right)\right)-(-2y-2\pi e^{-\frac{2\pi^2 t}{Re}}((cos\pi x-sin\,\pi x)sin\,\pi y))(Re(y-\pi e^{-\frac{2\pi^2 t}{Re}}((sin\pi x+cos\,\pi x)sin\,\pi y)))}{(Re(100+xy+e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x-sin\,\pi x)sin\,\pi y\right))^2} \quad (4.18)$$

$$u_y = \frac{-2-2\pi^2 e^{-\frac{2\pi^2 t}{Re}}\left((sin\pi x+cos\,\pi x)\cos\pi y\right)\left(Re\left(100+xy+e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x-sin\,\pi x)sin\,\pi y\right)\right)\right)-(-2y-2\pi e^{-\frac{2\pi^2 t}{Re}}((cos\pi x-sin\,\pi x)sin\,\pi y))(Re(x-\pi e^{-\frac{2\pi^2 t}{Re}}((sin\pi x-cos\,\pi x)cos\,\pi y)))}{(Re(100+xy+e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x-sin\,\pi x)sin\,\pi y\right))^2} \quad (4.19)$$

$$v_x = \frac{-2-2\pi^2 e^{-\frac{2\pi^2 t}{Re}}\left((sin\pi x+cos\,\pi x)\cos\pi y\right)\left(Re\left(100+xy+e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x-sin\,\pi x)sin\,\pi y\right)\right)\right)-(-2x-2\pi e^{-\frac{2\pi^2 t}{Re}}((cos\pi x-sin\,\pi x)sin\,\pi y))(Re(y-\pi e^{-\frac{2\pi^2 t}{Re}}((sin\pi x+cos\,\pi x)sin\,\pi y)))}{(Re(100+xy+e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x-sin\,\pi x)sin\,\pi y\right))^2} \quad (4.20)$$

$$v_y = \frac{-2\pi^2 e^{-\frac{2\pi^2 t}{Re}}\left((cos\,\pi x+sin\pi x)\sin\pi y\right)\left(Re\left(100+xy+e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x-sin\,\pi x)sin\,\pi y\right)\right)\right)-(-2x-2\pi e^{-\frac{2\pi^2 t}{Re}}((cos\pi x-sin\,\pi x)sin\,\pi y))(Re(x-\pi e^{-\frac{2\pi^2 t}{Re}}((sin\pi x-cos\,\pi x)cos\,\pi y)))}{(Re(100+xy+e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x-sin\,\pi x)sin\,\pi y\right))^2} \quad (4.21)$$

Using forward finite difference to approximate equations (4.21) gives:

$$\frac{U_{m+1,l,\omega} - U_{m,l,\omega}}{h}$$

$$2\pi^2 e^{-\frac{2\pi^2 t}{Re}}\left((sin\pi x + cos\,\pi x)\,sin\,\pi y\right)\left(Re\left(100 + xy + e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x - sin\,\pi x)sin\,\pi y\right)\right)\right) -$$

$$= \frac{(-2y - 2\pi e^{-\frac{2\pi^2 t}{Re}}((cos\pi x - sin\,\pi x)sin\,\pi y))(Re(y - \pi e^{-\frac{2\pi^2 t}{Re}}((sin\pi x + cos\,\pi x)sin\,\pi y)))}{(Re(100 + xy + e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x - sin\,\pi x)sin\,\pi y\right))^2}$$

(4.22)

$$\frac{U_{m,l+1,\omega} - U_{m,l,\omega}}{q} = \frac{-2 - 2\pi^2 e^{-\frac{2\pi^2 t}{Re}}\left((sin\pi x + cos\,\pi x)\cos\pi y\right)\left(Re\left(100 + xy + e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x - sin\,\pi x)sin\,\pi y\right)\right)\right) -}{(-2y - 2\pi e^{-\frac{2\pi^2 t}{Re}}((cos\pi x - sin\,\pi x)sin\,\pi y))(Re(x - \pi e^{-\frac{2\pi^2 t}{Re}}((sin\pi x - cos\,\pi x)cos\,\pi y)))}{(Re(100 + xy + e^{-\frac{2\pi^2 t}{Re}}\left((cos\pi x - sin\,\pi x)sin\,\pi y\right))^2}$$

(4.23)

We use Newman's boundary conditions at the boundaries to approximate $U_{m\pm2,l,\omega}$ and $U_{m,l\pm2,\omega}$

At the boundaries $x = 0$ and $y = 0$, we have

$$\frac{U_{m-1,l,\omega} - U_{m,l-2,\omega}}{h} = \pi\, e^{-\pi^2\,\omega\,k}\,(0)$$

$$\frac{U_{m,l-1,\omega} - U_{m,l-2,\omega}}{q} = \pi\, e^{-\pi^2\,\omega\,k}\,(0) \quad \text{respectively} \tag{4.24}$$

And so

$$U_{m-2,l,\omega} = U_{m-1,l,\omega}$$

$$U_{m,l-2,\omega} = U_{m,l-1,\omega} \tag{4.25}$$

At the boundaries $x = 1$ and $y = 1$, we have

$$\frac{U_{m+2,l,\omega} - U_{m,l+1,\omega}}{h} = \pi\, e^{-\pi^2\,\omega\,k}\,(0)$$

$$\frac{U_{m,l+2,\omega} - U_{m,l+1,\omega}}{q} = \pi\, e^{-\pi^2\, \omega\, k}\,(0) \text{ respectively} \tag{4.26}$$

and so

$$U_{m+2,l,\omega} = U_{m+1,l,\omega}$$

$$U_{m,l+2,\omega} = U_{m,l+1,\omega} \tag{4.27}$$

In equations (4.22)-(4.27) $\omega = n$ or $n+1$

For $\omega = n+1$

$$U_{m-2,l,n+1} = U_{m-1,l,n+1}$$

$$U_{m,l-2,n+1} = U_{m,l-1,n+1} \tag{4.28}$$

and

$$U_{m+2,l,n+1} = U_{m+1,l,n+1}$$

$$U_{m,l+2,n+1} = U_{m,l+1,n+1} \tag{4.29}$$

For $\omega = n$

$$U_{m-2,l,n} = U_{m-1,l,n}$$

$$U_{m,l-2,n} = U_{m,l-1,n} \tag{4.30}$$

and

$$U_{m+2,l,n} = U_{m+1,l,n}$$

$$U_{m,l+2,n} = U_{m,l+1,n} \tag{4.31}$$

Using equations (4.28)-(4.31) in equation (4.15) we obtain the pure Crank-Nicholson scheme as shown below

$$\big(1 + 2kaU_{m,l,n} + 2kaV_{m,l,n} - 12kb - 4k^2a^2U_{m,l,n}V_{m,l,n} + 32k^2abU_{m,l,n} + 32k^2abV_{m,l,n}$$

$$- 36k^2b^2 - 6ka^2U^2{}_{m,l,n} - 6kb^2V^2{}_{m,l,n} - 140b^2\big)U_{m,l,n+1} +$$

$$\left(-kaU_{m,l,n}+3kb+2k^2a^2U_{m,l,n}V_{m,l,n}-16k^2abU_{m,l,n}-6k^2abV_{m,l,n}+18k^2b^2+\frac{3}{2}ka^2U^2{}_{m,l,n}+\frac{35}{2}b^2\right)U_{m+1,l,n+1}+\left(-kaV_{m,l,n}+3kb+2k^2a^2U_{m,l,n}V_{m,l,n}-16k^2abU_{m,l,n}-6k^2abV_{m,l,n}+18k^2b^2+\frac{3}{2}kb^2V^2{}_{m,l,n}+\frac{35}{2}b^2\right)U_{m,l+1,n+1}-\left(-k^2a^2U_{m,l,n}V_{m,l,n}+3k^2abU_{m,l,n}+3k^2abV_{m,l,n}-5k^2ab\right)U_{m+1,l+1,n+1}+\left(-kaU_{m,l,n}+3kb+2k^2a^2U_{m,l,n}V_{m,l,n}-16k^2abU_{m,l,n}-6k^2abV_{m,l,n}+18k^2ab+\frac{3}{2}ka^2U^2{}_{m,l,n}+\frac{35}{2}kb^2\right)U_{m-1,l,n+1}+\left(-kaV_{m,l,n}+3kb+2k^2a^2U_{m,l,n}V_{m,l,n}-6k^2abU_{m,l,n}-16k^2abV_{m,l,n}-6k^2b^2+\frac{3}{2}kb^2V^2{}_{m,l,n}+\frac{35}{2}b^2\right)U_{m,l-1,n+1}+\left(-k^2a^2U_{m,l,n}V_{m,l,n}+3k^2abU_{m,l,n}+3k^2abV_{m,l,n}-9k^2ab\right)U_{m-1,l+1,n+1}+\left(-k^2a^2U_{m,l,n}V_{m,l,n}+3k^2abU_{m,l,n}+3k^2abV_{m,l,n}-13k^2ab\right)U_{m+1,l-1,n+1}+\left(-k^2a^2U_{m,l,n}V_{m,l,n}+3k^2abU_{m,l,n}+3k^2abV_{m,l,n}-9k^2ab\right)U_{m-1,l-1,n+1}=-\left(2kaU_{m,l,n}+2kaV_{m,l,n}-12kb-4k^2a^2U_{m,l,n}V_{m,l,n}+32k^2abU_{m,l,n}+32k^2abV_{m,l,n}-36k^2b^2-6ka^2U^2{}_{m,l,n}-6kb^2V^2{}_{m,l,n}-140b^2\right)U_{m,l,n}-\left(-kaU_{m,l,n}+3kb+2k^2a^2U_{m,l,n}V_{m,l,n}-16k^2abU_{m,l,n}-6k^2abV_{m,l,n}+18k^2b^2+\frac{3}{2}ka^2U^2{}_{m,l,n}+\frac{35}{2}b^2\right)U_{m+1,l,n}-\left(-kaV_{m,l,n}+3kb+2k^2a^2U_{m,l,n}V_{m,l,n}-16k^2abU_{m,l,n}-6k^2abV_{m,l,n}+18k^2b^2+\frac{3}{2}kb^2V^2{}_{m,l,n}+\frac{35}{2}b^2\right)U_{m,l+1,n}-\left(-k^2a^2U_{m,l,n}V_{m,l,n}+3k^2abU_{m,l,n}+3k^2abV_{m,l,n}-5k^2ab\right)U_{m+1,l+1,n}-\left(-kaU_{m,l,n}+3kb+2k^2a^2U_{m,l,n}V_{m,l,n}-16k^2abU_{m,l,n}-6k^2abV_{m,l,n}+18k^2ab+\frac{3}{2}ka^2U^2{}_{m,l,n}+\frac{35}{2}kb^2\right)U_{m-1,l,n}-\left(-kaV_{m,l,n}+3kb+2k^2a^2U_{m,l,n}V_{m,l,n}-6k^2abU_{m,l,n}-16k^2abV_{m,l,n}-6k^2b^2+\frac{3}{2}kb^2V^2{}_{m,l,n}+\frac{35}{2}b^2\right)U_{m,l-1,n}-\left(-k^2a^2U_{m,l,n}V_{m,l,n}+3k^2abU_{m,l,n}+3k^2abV_{m,l,n}-\right.$$

$$9k^2ab\big)U_{m-1,l+1,n} - \big(-k^2a^2U_{m,l,n}V_{m,l,n} + 3k^2abU_{m,l,n} + 3k^2abV_{m,l,n} - 13k^2ab\big)U_{m+1,l-1,n} - \big(-k^2a^2U_{m,l,n}V_{m,l,n} + 3k^2abU_{m,l,n} + 3k^2abV_{m,l,n} - 9k^2ab\big)U_{m-1,l-1,n} \quad (4.32)$$

4.3 Crank-Nicholson-Du-Fort and Frankel (CN-DF) Scheme

The Crank-Nicholson-Du-Fort and Frankel Scheme is obtained by replacing $U_{m,l,n}$ by $\frac{1}{2}(U_{m,l,n+1} + U_{m,l,n-1})$ in equations (4.32) to give the CN-DF scheme below:

$$\Big(1 + ka\,(U_{m,l,n+1} + U_{m,l,n-1}) + 2kaV_{m,l,n} - 12kb - 2k^2a^2(U_{m,l,n+1} + U_{m,l,n-1})V_{m,l,n} + 8k^2ab\,(U_{m,l,n+1} + U_{m,l,n-1}) + 32k^2abV_{m,l,n} - 36k^2b^2 - \tfrac{3}{2}ka^2(U_{m,l,n+1} + U_{m,l,n-1})^2 - 6kb^2V^2{}_{m,l,n} - 140b^2\Big)U_{m,l,n+1} + \Big(-\tfrac{1}{2}ka(U_{m,l,n+1} + U_{m,l,n-1}) + 3kb + k^2a^2\,(U_{m,l,n+1} + U_{m,l,n-1})V_{m,l,n} - 8k^2\,ab(U_{m,l,n+1} + U_{m,l,n-1}) - 6k^2abV_{m,l,n} + 18k^2b^2 + \tfrac{3}{8}ka^2\,(U_{m,l,n+1} + U_{m,l,n-1})^2 + \tfrac{35}{2}b^2\Big)U_{m+1,l,n+1} + \Big(-kaV_{m,l,n} + 3kb + k^2a^2(U_{m,l,n+1} + U_{m,l,n-1})V_{m,l,n} - 16k^2abU_{m,l,n} - 6k^2abV_{m,l,n} + 18k^2b^2 + \tfrac{3}{2}kb^2V^2{}_{m,l,n} + \tfrac{35}{2}b^2\Big)U_{m,l+1,n+1} - \Big(-\tfrac{1}{2}k^2a^2(U_{m,l,n+1} + U_{m,l,n-1})V_{m,l,n} + \tfrac{3}{2}k^2(U_{m,l,n+1} + U_{m,l,n-1}) + 3k^2abV_{m,l,n} - 5k^2ab\Big)U_{m+1,l+1,n+1} + \Big(-\tfrac{1}{2}ka(U_{m,l,n+1} + U_{m,l,n-1}) + 3kb + k^2a^2(U_{m,l,n+1} + U_{m,l,n-1})V_{m,l,n} - 8k^2ab(U_{m,l,n+1} + U_{m,l,n-1}) - 6k^2abV_{m,l,n} + 18k^2ab + \tfrac{3}{8}ka^2(U_{m,l,n+1} + U_{m,l,n-1}) + \tfrac{35}{2}kb^2\Big)U_{m-1,l,n+1} + \Big(-kaV_{m,l,n} + 3kb + a^2(U_{m,l,n+1} + U_{m,l,n-1})\;V_{m,l,n} - 3k^2ab(U_{m,l,n+1} + U_{m,l,n-1}) - 16k^2abV_{m,l,n} - 6k^2b^2 + \tfrac{3}{2}kb^2V^2{}_{m,l,n} + \tfrac{35}{2}b^2\Big)U_{m,l-1,n+1} + \Big(-\tfrac{1}{2}k^2a^2(U_{m,l,n+1} + U_{m,l,n-1})\;V_{m,l,n} + \tfrac{3}{2}k^2ab(U_{m,l,n+1} + U_{m,l,n-1}) + 3k^2abV_{m,l,n} - 9k^2ab\Big)U_{m-1,l+1,n+1} +$$

$$\left(-k^2a^2U_{m,l,n}V_{m,l,n}+\frac{3}{2}k^2ab(U_{m,l,n+1}\ +\ U_{m,l,n-1})\ \ +3k^2abV_{m,l,n}-13k^2ab\right)U_{m+1,l-1,n+1}+$$

$$\left(-k^2a^2U_{m,l,n}V_{m,l,n}+\frac{3}{2}k^2ab(U_{m,l,n+1}\ +\ U_{m,l,n-1})\ \ +3k^2abV_{m,l,n}-9k^2ab\right)U_{m-1,l-1,n+1}=$$

$$-\Big(ka\ (U_{m,l,n+1}\ +\ U_{m,l,n-1})+2kaV_{m,l,n}-12kb-2k^2a^2(U_{m,l,n+1}\ +\ U_{m,l,n-1})V_{m,l,n}+$$

$$8k^2ab\ (U_{m,l,n+1}\ +\ U_{m,l,n-1})+32k^2abV_{m,l,n}-36k^2b^2-\frac{3}{2}ka^2(U_{m,l,n+1}\ +\ U_{m,l,n-1})^2-$$

$$6kb^2{V^2}_{m,l,n}-140b^2\Big)U_{m,l,n}-\Big(-\frac{1}{2}ka(U_{m,l,n+1}\ +\ U_{m,l,n-1})+3kb+k^2a^2\ (U_{m,l,n+1}+$$

$$U_{m,l,n-1})V_{m,l,n}-8k^2\ ab(U_{m,l,n+1}\ +\ U_{m,l,n-1})-6k^2abV_{m,l,n}+18k^2b^2+\frac{3}{8}ka^2\ (U_{m,l,n+1}+$$

$$U_{m,l,n-1})^2+\frac{35}{2}b^2\Big)U_{m+1,l,n}-\Big(-kaV_{m,l,n}+3kb+k^2a^2(U_{m,l,n+1}\ +\ U_{m,l,n-1})V_{m,l,n}-$$

$$16k^2abU_{m,l,n}-6k^2abV_{m,l,n}+18k^2b^2+\frac{3}{2}kb^2{V^2}_{m,l,n}+\frac{35}{2}b^2\Big)U_{m,l+1,n}-$$

$$\Big(-\frac{1}{2}k^2a^2(U_{m,l,n+1}\ +\ U_{m,l,n-1})V_{m,l,n}+\frac{3}{2}k^2(U_{m,l,n+1}\ +\ U_{m,l,n-1})+3k^2abV_{m,l,n}-$$

$$5k^2ab\Big)U_{m+1,l+1,n}-$$

$$\Big(-\frac{1}{2}ka(U_{m,l,n+1}\ +\ U_{m,l,n-1})+3kb+k^2a^2(U_{m,l,n+1}\ +\ U_{m,l,n-1})V_{m,l,n}-8k^2ab(U_{m,l,n+1}+$$

$$U_{m,l,n-1})-6k^2abV_{m,l,n}+18k^2ab+\frac{3}{8}ka^2(U_{m,l,n+1}\ +\ U_{m,l,n-1})+\frac{35}{2}kb^2\Big)U_{m-1,l,n}+$$

$$\Big(-kaV_{m,l,n}+3kb+a^2(U_{m,l,n+1}\ +\ U_{m,l,n-1})\ \ V_{m,l,n}-3k^2ab(U_{m,l,n+1}\ +\ U_{m,l,n-1})\ \ -$$

$$16k^2abV_{m,l,n}-6k^2b^2+\frac{3}{2}kb^2{V^2}_{m,l,n}+\frac{35}{2}b^2\Big)U_{m,l-1,n}+$$

$$\Big(-\frac{1}{2}k^2a^2(U_{m,l,n+1}\ +\ U_{m,l,n-1})\ \ V_{m,l,n}+\frac{3}{2}k^2ab(U_{m,l,n+1}\ +\ U_{m,l,n-1})\ \ +3k^2abV_{m,l,n}-$$

$$9k^2ab\Big)U_{m-1,l+1,n}+\Big(-k^2a^2U_{m,l,n}V_{m,l,n}+\frac{3}{2}k^2ab(U_{m,l,n+1}\ +\ U_{m,l,n-1})\ \ +3k^2abV_{m,l,n}-$$

$$13k^2ab\Big)U_{m+1,l-1,n}+\Big(-k^2a^2U_{m,l,n}V_{m,l,n}+\frac{3}{2}k^2ab(U_{m,l,n+1}\ +\ U_{m,l,n-1})\ \ +3k^2abV_{m,l,n}-$$

$$9k^2ab\Big)U_{m-1,l-1,n} \qquad (4.33)$$

4.4 Crank-Nicholson-Lax-Friedrich's (CN-LF) Scheme

For this scheme the first term $U_{m,l,n}$ in the right hand side of equation (4.32) is replaced by $\frac{1}{2}\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \frac{1}{2}\left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)$ to get:

$$\left(1 + ka\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) + 2kaV_{m,l,n} - 12kb - 2k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) + 16k^2\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \frac{1}{2}\left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right) + 32k^2abV_{m,l,n} - 36k^2b^2 - \frac{3}{2}ka^2\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right)^2 - 6kb^2V^2{}_{m,l,n} - 140b^2\right)U_{m,l,n+1} +$$

$$\left(-\frac{1}{2}ka\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) + 3kb + k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) - 8k^2ab\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) - 6k^2abV_{m,l,n} + 18k^2b^2 + \frac{3}{8}ka^2\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right)^2 + \frac{35}{2}b^2\right)U_{m+1,l,n+1} + \left(-kaV_{m,l,n} + 3kb + k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) - 8k^2ab\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) - 6k^2abV_{m,l,n} + 18k^2b^2 + \frac{3}{2}kb^2V^2{}_{m,l,n} + \frac{35}{2}b^2\right)U_{m,l+1,n+1} -$$

$$\left(-\frac{1}{2}k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) + \frac{3}{2}k^2ab\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) + 3k^2abV_{m,l,n} - 5k^2ab\right)U_{m+1,l+1,n+1} + \left(-\frac{1}{2}ka\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) + 3kb + k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n} + U_{m-1,l+1,n}\right) + \left(U_{m+1,l-1,n} + U_{m-1,l-1,n}\right)\right) -$$

$$8k^2ab\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)-6k^2abV_{m,l,n}+18k^2ab+$$

$$\frac{3}{8}ka^2\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)^2+\frac{35}{2}kb^2\Big)U_{m-1,l,n+1}+$$

$$\left(-kaV_{m,l,n}+3kb+k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)-\right.$$

$$3k^2ab\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)U_{m,l,n}-16k^2abV_{m,l,n}-$$

$$6k^2b^2+\frac{3}{2}kb^2{V^2}_{m,l,n}+\frac{35}{2}b^2\Big)U_{m,l-1,n+1}+\left(-\frac{1}{2}k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\right.\right.$$

$$\left.\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)+\frac{3}{2}k^2ab\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\right.\right.$$

$$\left.\left.U_{m-1,l-1,n}\right)\right)+3k^2abV_{m,l,n}-9k^2ab\Big)U_{m-1,l+1,n+1}+\left(-\frac{1}{2}k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n}\ +\right.\right.\right.$$

$$\left.\left.\left.U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)+\frac{3}{2}k^2ab\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\right.\right.$$

$$\left.\left.\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)+3k^2abV_{m,l,n}-13k^2ab\Big)U_{m+1,l-1,n+1}+$$

$$\left(-\frac{1}{2}k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)+\right.$$

$$\frac{3}{2}k^2ab\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)+3k^2abV_{m,l,n}-$$

$$9k^2ab\Big)U_{m-1,l-1,n+1}=-\frac{1}{2}\Big(ka\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)+$$

$$2kaV_{m,l,n}-12kb-2k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)+$$

$$16k^2\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\frac{1}{2}\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)+32k^2abV_{m,l,n}-36k^2b^2-$$

$$\frac{3}{2}ka^2\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)^2-6kb^2{V^2}_{m,l,n}-$$

$$140b^2\Big)\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)+\left(-\frac{1}{2}ka\left(\left(U_{m+1,l+1,n}\ +\right.\right.\right.$$

$$\left.\left.\left.U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)+3kb+k^2a^2V_{m,l,n}\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\right.\right.$$

$$\left.\left.\left(U_{m+1,l-1,n}\ +\ U_{m-1,l-1,n}\right)\right)-8k^2ab\left(\left(U_{m+1,l+1,n}\ +\ U_{m-1,l+1,n}\right)+\left(U_{m+1,l-1,n}\ +\right.\right.\right.$$

$$U_{m-1,l-1,n}\Big)\Big) - 6k^2abV_{m,l,n} + 18k^2b^2 + \frac{3}{8}ka^2\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} +$$

$$U_{m-1,l-1,n}\big)\Big)^2 + \frac{35}{2}b^2\Big)U_{m+1,l,n} + \Big(-kaV_{m,l,n} + 3kb + k^2a^2V_{m,l,n}\Big(\big(U_{m+1,l+1,n} +$$

$$U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) - 8k^2ab\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) +$$

$$\big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) - 6k^2abV_{m,l,n} + 18k^2b^2 + \frac{3}{2}kb^2V^2{}_{m,l,n} + \frac{35}{2}b^2\Big)U_{m,l+1,n} -$$

$$\Big(-\frac{1}{2}k^2a^2V_{m,l,n}\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) +$$

$$\frac{3}{2}k^2ab\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) + 3k^2abV_{m,l,n} -$$

$$5k^2ab\Big)U_{m+1,l+1,n} + \Big(-\frac{1}{2}ka\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) +$$

$$3kb + k^2a^2V_{m,l,n}\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) -$$

$$8k^2ab\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) - 6k^2abV_{m,l,n} + 18k^2ab +$$

$$\frac{3}{8}ka^2\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big)^2 + \frac{35}{2}kb^2\Big)U_{m-1,l,n} +$$

$$\Big(-kaV_{m,l,n} + 3kb + k^2a^2V_{m,l,n}\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) -$$

$$3k^2ab\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) \quad - 16k^2abV_{m,l,n} - 6k^2b^2 +$$

$$\frac{3}{2}kb^2V^2{}_{m,l,n} + \frac{35}{2}b^2\Big)U_{m,l-1,n} + \Big(-\frac{1}{2}k^2a^2V_{m,l,n}\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} +$$

$$U_{m-1,l-1,n}\big)\Big) + \frac{3}{2}k^2ab\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) +$$

$$3k^2abV_{m,l,n} - 9k^2ab\Big)U_{m-1,l+1,n} + \Big(-\frac{1}{2}k^2a^2V_{m,l,n}\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) +$$

$$\big(U_{m+1,l-1,n} + U_{m-1,l-1,n}\big)\Big) + \frac{3}{2}k^2ab\Big(\big(U_{m+1,l+1,n} + U_{m-1,l+1,n}\big) + \big(U_{m+1,l-1,n} +$$

$$U_{m-1,l-1,n}\big)\Big) + 3k^2abV_{m,l,n} - 13k^2ab\Big)U_{m+1,l-1,n} + \Big(-\frac{1}{2}k^2a^2V_{m,l,n}\Big(\big(U_{m+1,l+1,n} +$$

$$U_{m-1,l+1,n}) + (U_{m+1,l-1,n} + U_{m-1,l-1,n})\Big) + \frac{3}{2}k^2ab\Big((U_{m+1,l+1,n} + U_{m-1,l+1,n}) + (U_{m+1,l-1,n} + U_{m-1,l-1,n})\Big) + 3k^2abV_{m,l,n} - 9k^2ab\Big)U_{m-1,l-1,n}$$

(4.34)

4.5 Crank-Nicholson-Du-Fort and Frankel -Lax-Friedrich's (CN-DF-LF) Scheme

In the equation (4.34) is modified replacing $U_{m,l,n-1}$ by $\frac{1}{2}(U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + \frac{1}{2}(U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})$ to get:

$$\Big(1 + ka\,(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) + 2kaV_{m,l,n} - 12kb - 2k^2a^2(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big))V_{m,l,n} + 8k^2ab\,(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) + 32k^2abV_{m,l,n} - 36k^2b^2 - \frac{3}{2}ka^2(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big))^2 - 6kb^2V^2{}_{m,l,n} - 140b^2\Big)U_{m,l,n+1} + \Big(-\frac{1}{2}ka(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) + 3kb + k^2a^2\,(U_{m,l,n+1} + U_{m,l,n-1})V_{m,l,n} - 8k^2\,ab(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) - 6k^2abV_{m,l,n} + 18k^2b^2 + \frac{3}{8}ka^2\,(U_{m,l,n+1} + U_{m,l,n-1})^2 + \frac{35}{2}b^2\Big)U_{m+1,l,n+1} + \Big(-kaV_{m,l,n} + 3kb + k^2a^2(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big))V_{m,l,n} - 16k^2abU_{m,l,n} - 6k^2abV_{m,l,n} + 18k^2b^2 + \frac{3}{2}kb^2V^2{}_{m,l,n} + \frac{35}{2}b^2\Big)U_{m,l+1,n+1} - \Big(-\frac{1}{2}k^2a^2(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big))V_{m,l,n} + \frac{3}{2}k^2(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) + 3k^2abV_{m,l,n} - 5k^2ab\Big)U_{m+1,l+1,n+1} + \Big(-\frac{1}{2}ka(U_{m,l,n+1} +$$

$$U_{m,l,n-1}) + 3kb + k^2a^2(U_{m,l,n+1} + U_{m,l,n-1})V_{m,l,n} - 8k^2ab(U_{m,l,n+1} + U_{m,l,n-1}) -$$

$$6k^2abV_{m,l,n} + 18k^2ab + \frac{3}{8}ka^2(U_{m,l,n+1} +$$

$$\frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) + \frac{35}{2}kb^2\Big)U_{m-1,l,n+1} +$$

$$\Big(-kaV_{m,l,n} + 3kb + a^2(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} +$$

$$U_{m-1,l-1,n-1})\big))V_{m,l,n} - 3k^2ab(U_{m,l,n+1} +$$

$$\frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) - 16k^2abV_{m,l,n} -$$

$$6k^2b^2 + \frac{3}{2}kb^2V^2{}_{m,l,n} + \frac{35}{2}b^2\Big)U_{m,l-1,n+1} + \Big(-\frac{1}{2}k^2a^2(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} +$$

$$U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big))V_{m,l,n} + \frac{3}{2}k^2ab(U_{m,l,n+1} +$$

$$\frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) + 3k^2abV_{m,l,n} -$$

$$9k^2ab\Big)U_{m-1,l+1,n+1} + \Big(-k^2a^2U_{m,l,n}V_{m,l,n} + \frac{3}{2}k^2ab(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} +$$

$$U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) + 3k^2abV_{m,l,n} - 13k^2ab\Big)U_{m+1,l-1,n+1} +$$

$$\Big(-k^2a^2U_{m,l,n}V_{m,l,n} + \frac{3}{2}k^2ab(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} +$$

$$U_{m-1,l-1,n-1})\big)) + 3k^2abV_{m,l,n} - 9k^2ab\Big)U_{m-1,l-1,n+1} = -\Big(ka\,(U_{m,l,n+1} +$$

$$\frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) + 2kaV_{m,l,n} - 12kb -$$

$$2k^2a^2(U_{m,l,n+1} +$$

$$\frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big))V_{m,l,n} +$$

$$8k^2ab\,(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)) +$$

$$32k^2abV_{m,l,n} - 36k^2b^2 - \frac{3}{2}ka^2(U_{m,l,n+1} + \frac{1}{2}\big((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) +$$

$$(U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\big)^2 - 6kb^2V^2{}_{m,l,n} - 140b^2\Big)U_{m,l,n} - \Big(-\frac{1}{2}ka(U_{m,l,n+1} +$$

$$\frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\right)) + 3kb +$$

$$k^2a^2\,(U_{m,l,n+1} +$$

$$\frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\right))V_{m,l,n} -$$

$$8k^2\,ab(U_{m,l,n+1} + \frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\right)) -$$

$$6k^2abV_{m,l,n} + 18k^2b^2 + \frac{3}{8}ka^2\,(U_{m,l,n+1} +$$

$$\frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\right))^2 + \frac{35}{2}b^2\Big)\,U_{m+1,l,n} -$$

$$\Big(-kaV_{m,l,n} + 3kb + k^2a^2(U_{m,l,n+1} + \frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} +$$

$$U_{m-1,l-1,n-1})\right))V_{m,l,n} - 16k^2abU_{m,l,n} - 6k^2abV_{m,l,n} + 18k^2b^2 + \frac{3}{2}kb^2{V^2}_{m,l,n} +$$

$$\frac{35}{2}b^2\Big)\,U_{m,l+1,n} - \Big(-\frac{1}{2}k^2a^2(U_{m,l,n+1} + \frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} +$$

$$U_{m-1,l-1,n-1})\right))V_{m,l,n} + \frac{3}{2}k^2(U_{m,l,n+1} + \frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} +$$

$$U_{m-1,l-1,n-1})\right)) + 3k^2abV_{m,l,n} - 5k^2ab\Big)\,U_{m+1,l+1,n} - \Big(-\frac{1}{2}ka(U_{m,l,n+1} + U_{m,l,n-1}) + 3kb +$$

$$k^2a^2(U_{m,l,n+1} +$$

$$\frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\right))V_{m,l,n} -$$

$$8k^2ab(U_{m,l,n+1} + \frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\right)) -$$

$$6k^2abV_{m,l,n} + 18k^2ab + \frac{3}{8}ka^2(U_{m,l,n+1} +$$

$$\frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1})\right)) + \frac{35}{2}kb^2\Big)\,U_{m-1,l,n} +$$

$$\Big(-kaV_{m,l,n} + 3kb + a^2(U_{m,l,n+1} + \frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} +$$

$$U_{m-1,l-1,n-1})\right)\;\; V_{m,l,n} - 3k^2ab(U_{m,l,n+1} + U_{m,l,n-1}) \;\; - 16k^2abV_{m,l,n} - 6k^2b^2 +$$

$$\frac{3}{2}kb^2{V^2}_{m,l,n} + \frac{35}{2}b^2\Big)\,U_{m,l-1,n} + \Big(-\frac{1}{2}k^2a^2(U_{m,l,n+1} + \frac{1}{2}\left((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) +\right.$$

$$(U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1}))) \; V_{m,l,n} + \frac{3}{2}k^2ab(U_{m,l,n+1} + \frac{1}{2}((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1}))) + 3k^2abV_{m,l,n} - 9k^2ab\Big)U_{m-1,l+1,n} + \Big(-k^2a^2U_{m,l,n}V_{m,l,n} + \frac{3}{2}k^2ab(U_{m,l,n+1} + \frac{1}{2}((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1}))) + 3k^2abV_{m,l,n} - 13k^2ab\Big)U_{m+1,l-1,n} + \Big(-k^2a^2U_{m,l,n}V_{m,l,n} + \frac{3}{2}k^2ab(U_{m,l,n+1} + \frac{1}{2}((U_{m+1,l+1,n-1} + U_{m-1,l+1,n-1}) + (U_{m+1,l-1,n-1} + U_{m-1,l-1,n-1}))) + 3k^2abV_{m,l,n} - 9k^2ab\Big)U_{m-1,l-1,n} \tag{4.35}$$

4.6 Numerical Solutions

In this section a presentation and an analysis of the numerical solution is made. Also presentation and discussion on the results obtained from the methods developed in chapter three is given. The following data is used:

$k = 0.001, h = 0.1, l = 0.1$ and $\mathrm{Re} = 5000$

Here the computational domain is taken as a square domain

$\propto = \{(x, y): 0 \leq x \leq 1, 0 \leq y \leq 1\}$. The initial and boundary conditions for $u(x, y, t)$and $v(x, y, t)$ are taken from the analytical solutions by (Kweyu, 2012).

The numerical computations are performed using uniform grid, with a mesh width $\Delta x = \Delta y = 0.1$

The results from schemes developed were generated using MATLAB and displayed using graphs, tables as well as three-dimensional figures.

Table 4.1: Solution of *u* for the 2-D Burgers equation for the different schemes

x	y	Kweyu *et al.* (2012) (e-0.006)	PURE CN (e-0.006)	CN-LF (e-0.006)	CN-DF (e-0.006)	CN-LF-DF (e-0.006)
0.1	0.1	-0.361650734301019	-0.361787155384327	-0.361637065349318	-0.361653771182265	-0.361652252771801
0.2	0.2	-0.725321995372639	-0.725588362525353	-0.725295305951890	-0.725327925052115	-0.725324960271592
0.3	0.3	-1.090398562803820	-1.090790321153970	-1.090359309108970	-1.090407283922360	-1.090402923450570
0.4	0.4	-1.455935883797210	-1.456451500936150	-1.455884219315910	-1.455947362252530	-1.455941623140330
0.5	0.5	-1.820803748461820	-1.821445349043890	-1.820739460401380	-1.820818031533110	-1.820810890141280
0.6	0.6	-2.183867594804460	-2.184640978240830	-2.183790102318200	-2.183884811546400	-2.183876203348610
0.7	0.7	-2.544179042412540	-2.545093045825040	-2.544087460418200	-2.544199389476210	-2.544189216148460
0.8	0.8	-2.901144228647610	-2.902209524822090	-2.901037488242520	-2.901167943526820	-2.901156086324040
0.9	0.9	-3.254642313537360	-3.255869840866490	-3.254519319265860	-3.254669639626780	-3.254655976853440
1	1	-3.605076050695350	-3.606475329816040	-3.604935849144370	-3.605107199831220	-3.605091625570710

Table 4.1 above shows the solutions of $\boldsymbol{u(x,y)}$ varying with values of $\boldsymbol{x}$ and $\boldsymbol{y}$. It is seen that the solutions for various schemes developed are closer to those of Kweyu *et al.* (2012) proposed solution of the two dimensional coupled Burgers' equation after Hopf-Cole transformation. The solutions are thus convergent for fixed value of $\boldsymbol{t}$ at $\boldsymbol{t = 1.0}$ and a fixed Reynolds number $\boldsymbol{Re = 5000}$.

Table 4.2: Solution of *v* for the 2-D Burgers equation for the different schemes

x	y	Kweyu *et al.* (2012) (e-0.006)	PURE CN (e-0.006)	CN-LF (e-0.006)	CN-DF (e-0.006)	CN-LF-DF (e-0.006)
0.1	0.1	-3.972769188311190	-3.972865925156520	-3.972759495591890	-3.972771341778260	-3.972770265066110
0.2	0.2	-3.944170064848750	-3.944368960551150	-3.944150135923460	-3.944174492525940	-3.944172278731560
0.3	0.3	-3.913141335562600	-3.913451475311220	-3.913110259949160	-3.913148239730720	-3.913144787715920
0.4	0.4	-3.878873375785220	-3.879306517566780	-3.878829975277790	-3.878883018208280	-3.878878197093740
0.5	0.5	-3.840895689849290	-3.841464954144770	-3.840838649848930	-3.840908362597330	-3.840902026350910
0.6	0.6	-3.799126642198930	-3.799845044283970	-3.799054658806100	-3.799142634968780	-3.799134638744720
0.7	0.7	-3.753877164071580	-3.754756190904300	-3.753789086686630	-3.753896732505020	-3.753886948484580
0.8	0.8	-3.705807681840330	-3.706856112188090	-3.705702631355120	-3.705831021264290	-3.705819351785390
0.9	0.9	-3.655845501933210	-3.657068620610160	-3.655722949394420	-3.655872729881250	-3.655859116177630
1	1	-3.605076050695350	-3.606475329816040	-3.604935849144370	-3.605107199831220	-3.605091625570710

Table 4.2 above shows the solutions of $\boldsymbol{v(x,y)}$ varying with values of $\boldsymbol{x}$ and $\boldsymbol{y}$. It is seen that the solutions for various schemes developed are closer to those of Kweyu *et al.* (2012) proposed solution of the two dimensional coupled Burgers' equation after Hopf-Cole transformation. The solutions are thus convergent for fixed value of $\boldsymbol{t}$ at $\boldsymbol{t = 1.0}$ and a fixed Reynolds number $\boldsymbol{Re = 5000}$.

We now present a table of absolute errors for a better comparison.

Table 4.3: Absolute Error in *u* for the 2-D Burgers equation for the different schemes

x	y	Pure CN (e-0.006)	CN-DF (e-0.006)	CN-LF (e-0.006)	CN-DF-LF (e-0.006)
0.1	0.1	0.000136421083307969	0.000003036881246	0.000013668951701	0.000001518470782
0.2	0.2	0.000266367152713998	0.000005929679476	0.000026689420749	0.000002964898953
0.3	0.3	0.000391758350150040	0.000008721118540	0.000039253694850	0.000004360646750
0.4	0.4	0.000515617138939994	0.000011478455320	0.000051664481300	0.000005739343120
0.5	0.5	0.000641600582069968	0.000014283071290	0.000064288060440	0.000007141679460
0.6	0.6	0.000773383436369901	0.000017216741940	0.000077492486260	0.000008608544150
0.7	0.7	0.000914003412499920	0.000020347063670	0.000091581994340	0.000010173735920
0.8	0.8	0.001065296174479700	0.000023714879210	0.000106740405090	0.000011857676430
0.9	0.9	0.001227527329129790	0.000027326089420	0.000122994271500	0.000013663316080
1	1	0.001399279120689820	0.000031149135870	0.000140201550980	0.000015574875360

It is noted in this Table 4.3 of absolute errors that as the values $\boldsymbol{x}$ increase from 0.1 to 1.0, the absolute errors in the numerical solutions of $\boldsymbol{u}(\boldsymbol{x},\boldsymbol{y})$ increases. The absolute errors in CN-DF-LF scheme are much smaller as compared to the other schemes. So it is most accurate, followed by CN-DF then CN-LF and lastly Pure CN scheme. This is on comparison with the Kweyu *et al.* (2012) proposed solution of the two dimensional coupled Burgers' equation after Hopf-Cole transformation.

Table 4.4: Absolute Error in *v* for the 2-D Burgers equation for the different schemes

x	y	Pure CN (e-0.006)	CN-DF (e-0.006)	CN-LF (e-0.006)	CN-DF-LF (e-0.006)
0.1	0.1	0.000096736845329737	0.000002153467070	0.000009692719300	0.000001076754920
0.2	0.2	0.000198895702399948	0.000004427677190	0.000019928925290	0.000002213882810
0.3	0.3	0.000310139748620042	0.000006904168120	0.000031075613440	0.000003452153320
0.4	0.4	0.000433141781559954	0.000009642423060	0.000043400507430	0.000004821308520
0.5	0.5	0.000569264295480210	0.000012672748040	0.000057040000360	0.000006336501620
0.6	0.6	0.000718402085040371	0.000015992769850	0.000071983392830	0.000007996545790
0.7	0.7	0.000879026832719898	0.000019568433440	0.000088077384950	0.000009784413000
0.8	0.8	0.001048430347760030	0.000023339423960	0.000105050485210	0.000011669945060
0.9	0.9	0.001223118676950020	0.000027227948040	0.000122552538790	0.000013614244420
1	1	0.001399279120689820	0.000031149135870	0.000140201550980	0.000015574875360

It is noted in this Table 4.4 of absolute errors that as the values $\boldsymbol{x}$ increase from 0.1 to 1.0, the absolute errors in the numerical solutions of $\boldsymbol{v}(\boldsymbol{x}, \boldsymbol{y})$ increases. The absolute errors in CN-DF-LF scheme are much smaller as compared to the other schemes. So it is most accurate, followed by CN-DF then CN-LF and lastly Pure CN scheme. This is on comparison with the Kweyu *et al.*, (2012) proposed solution of the two dimensional coupled Burgers' equation after Hopf-Cole transformation.

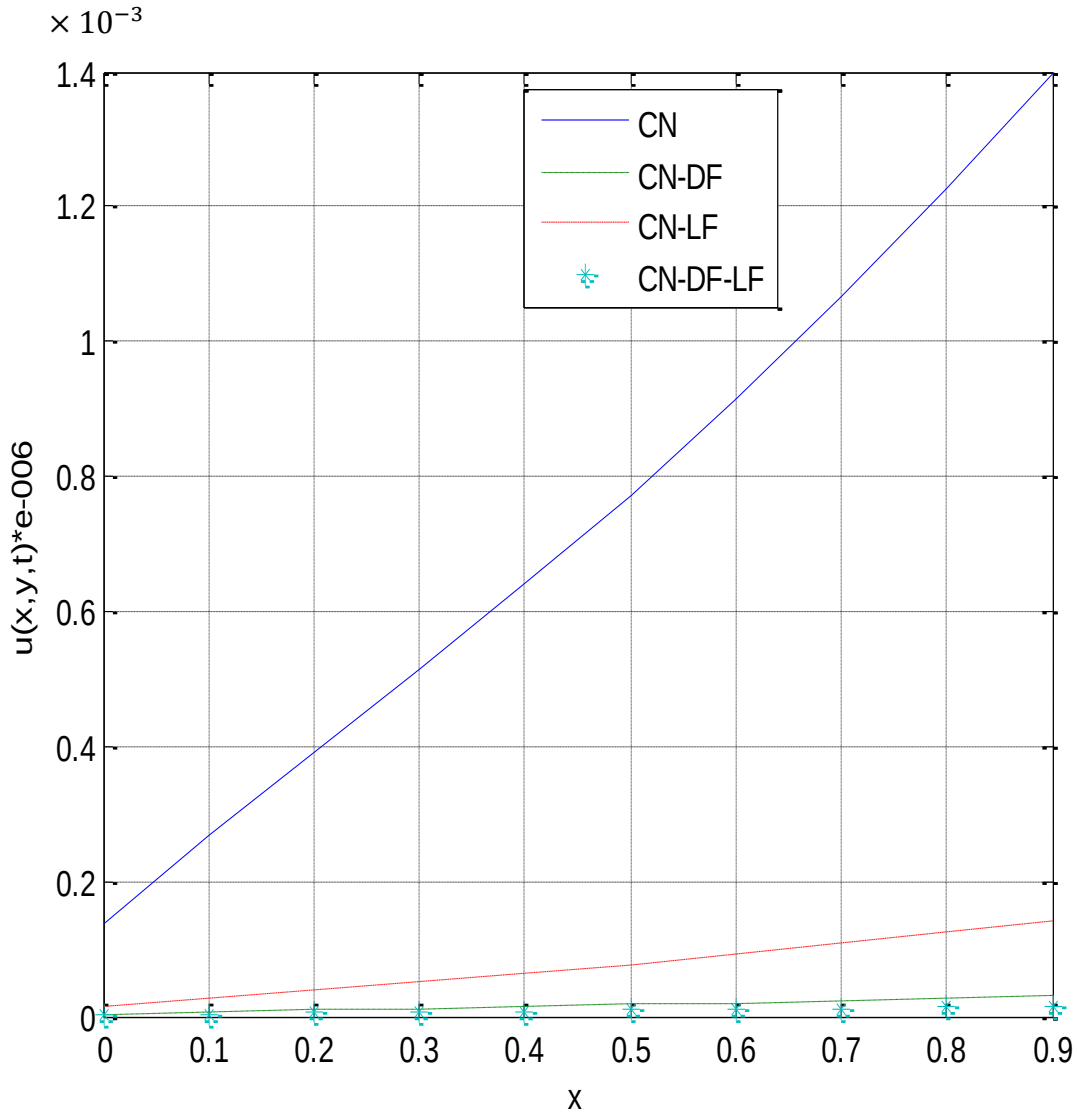

Figure 4.1: Absolute error in Solution of *u* for the 2-D Coupled Burgers' equation

The figure 4.1 of absolute errors above clearly shows that as the values $\boldsymbol{x}$ increase from 0.1 to 0.9, the absolute errors in the numerical solutions of $\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{y})$ increases. The absolute errors in CN-DF-LF scheme are much small as compared to the other schemes. So it is most accurate, followed by CN-DF then CN-LF and lastly Pure CN scheme being the least. This is on comparison with the Kweyu *et al.* (2012) proposed solution of the two dimensional coupled Burgers' equation after Hopf-Cole transformation.

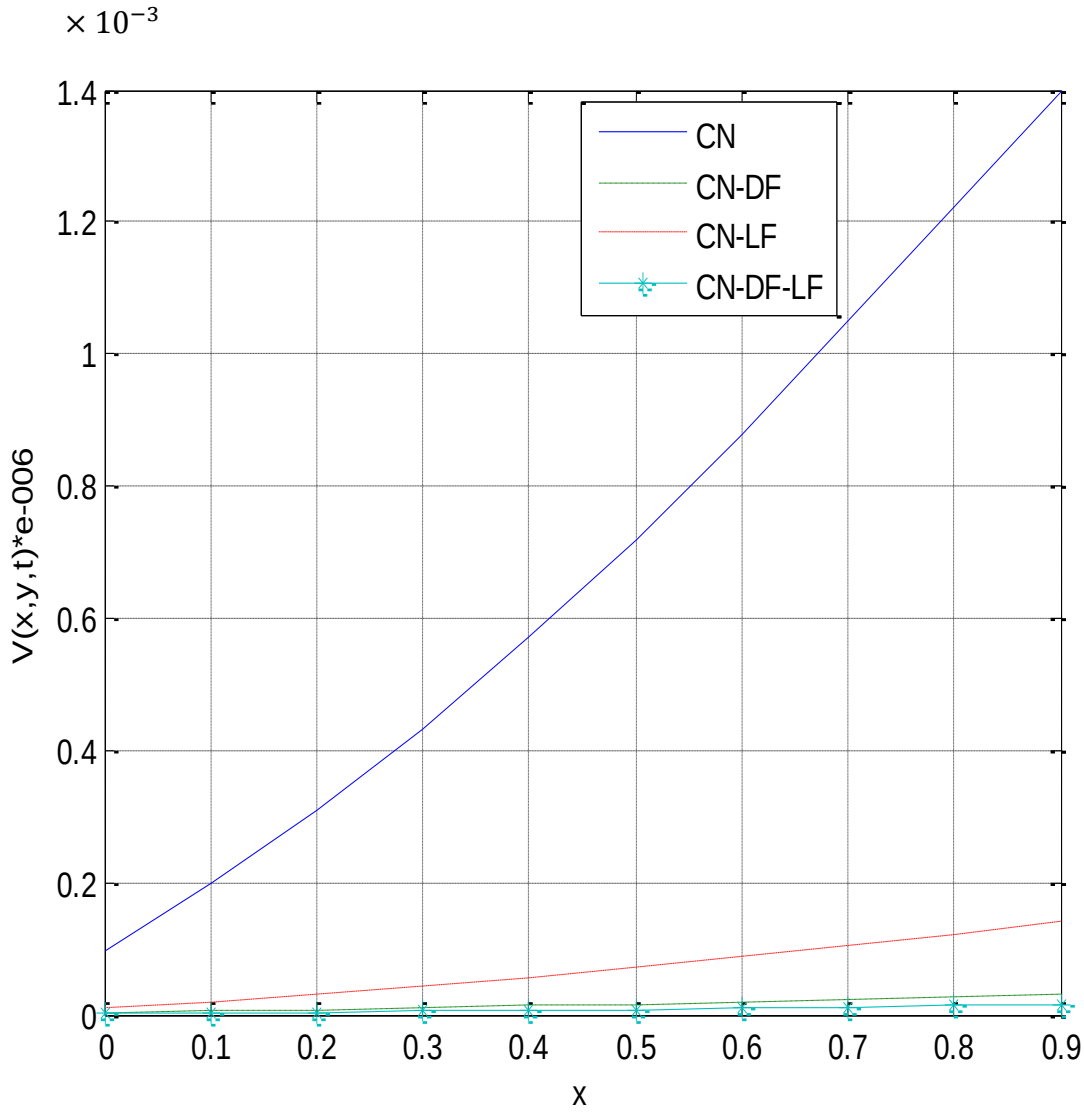

Figure 4.2: Absolute error in Solution of *v* for the 2-D Coupled Burgers' equation

The figure 4.2 of absolute errors above clearly shows that as the values $\boldsymbol{x}$ increase from 0.1 to 0.9, the absolute errors in the numerical solutions of $\boldsymbol{v}(\boldsymbol{x}, \boldsymbol{y})$ increases. The absolute errors in CN-DF-LF scheme are much small as compared to the other schemes. So it is most accurate, followed by CN-DF then CN-LF and lastly Pure CN scheme being the least. This is on comparison with the Kweyu *et al.*, (2012) proposed solution of the two dimensional coupled Burgers' equation after Hopf-Cole transformation.

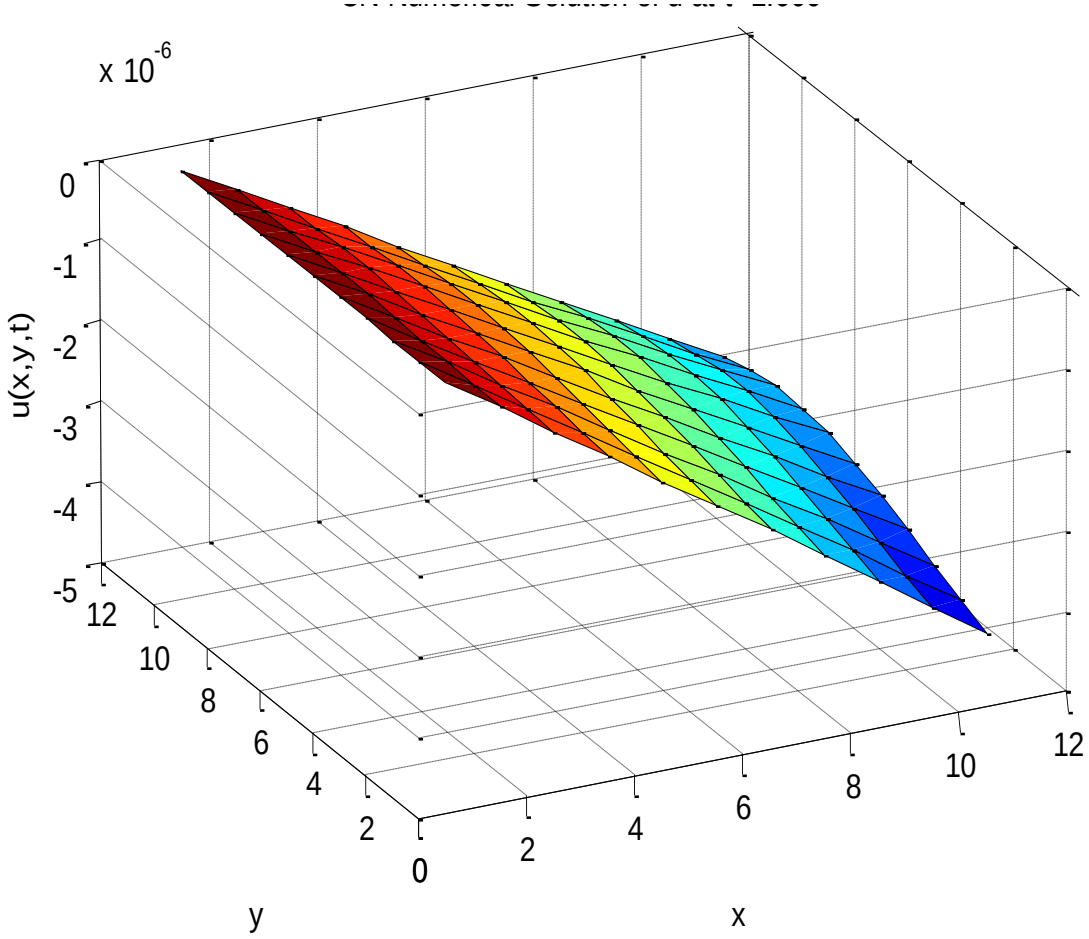

Figure 4.3 CN Numerical Solution of u

The variation of the solution of $\boldsymbol{u(x,y)}$ shown in the Figure 4.3 is a smooth curve and does not portray any sudden change for various values of $\boldsymbol{x}$ and $\boldsymbol{y}$ for the CN scheme. Thus the solutions from the developed scheme are consistent.

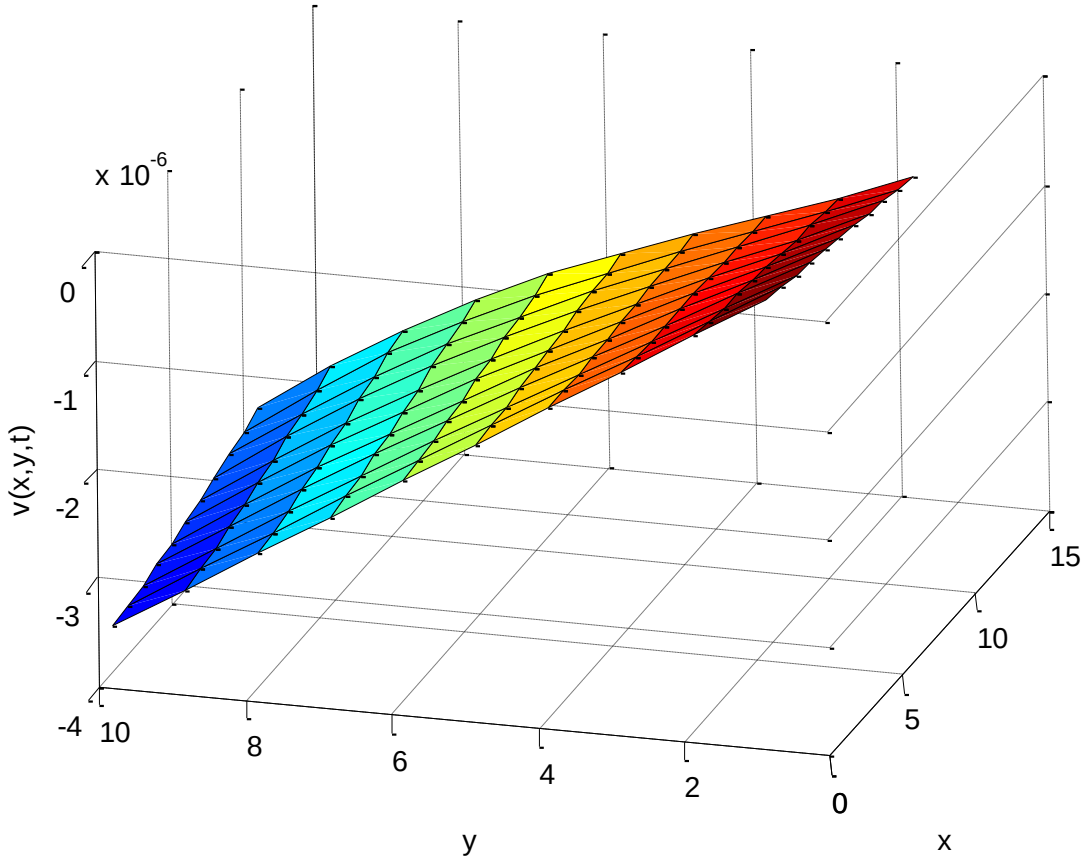

Figure 4.4: CN Numerical Solution of *v*

The variation of the solution of $\boldsymbol{v(x,y)}$ shown in the Figure 4.4 is a smooth curve and does not portray any sudden change for various values of $\boldsymbol{x}$ and $\boldsymbol{y}$ for the CN scheme. Thus the solutions from the developed scheme are consistent.

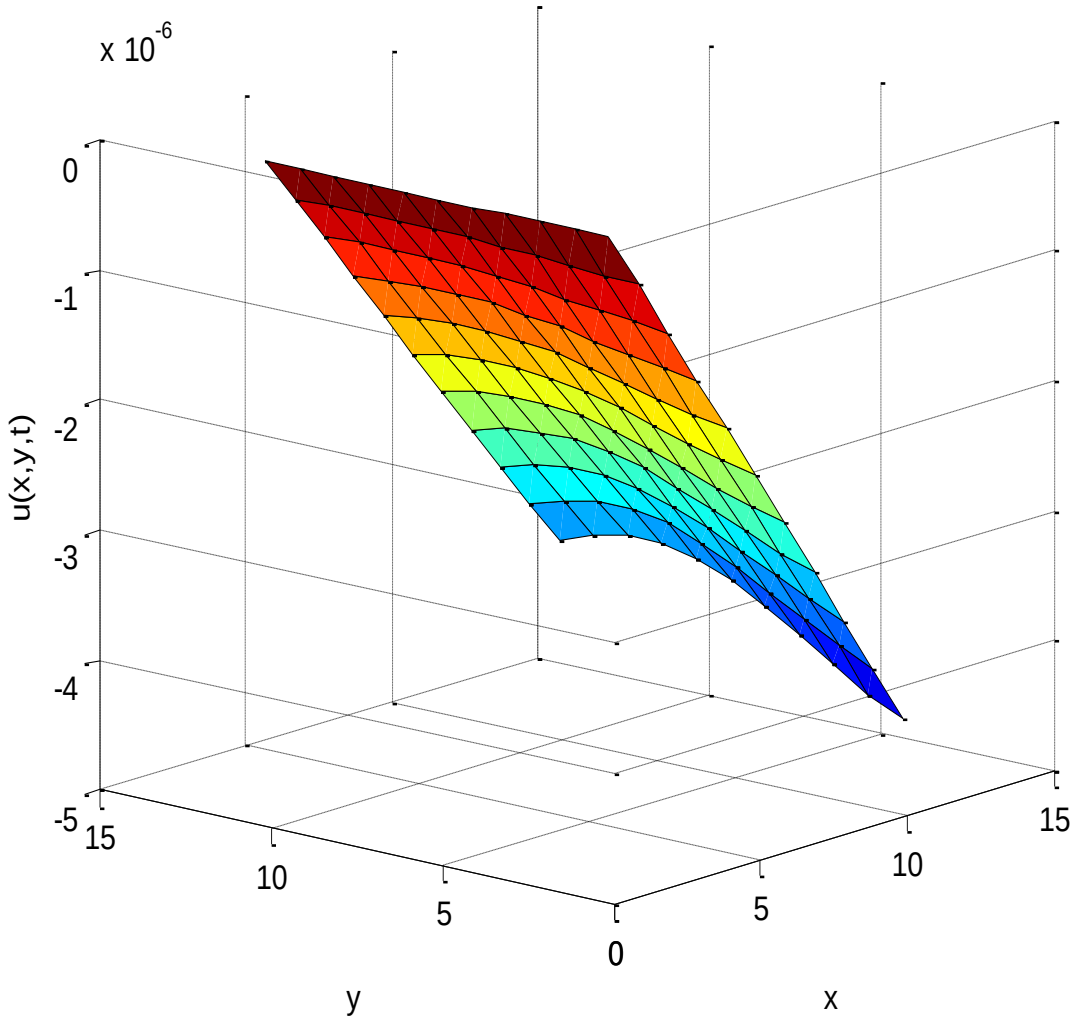

Figure 4.5: CN-LF Numerical Solution of u

The variation of the solution of $\boldsymbol{u(x,y)}$ shown in the Figure 4.5 is a smooth curve and does not portray any sudden change for various values of $\boldsymbol{x}$ and $\boldsymbol{y}$ for the CN-LF scheme. Thus the solutions from the developed scheme are consistent.

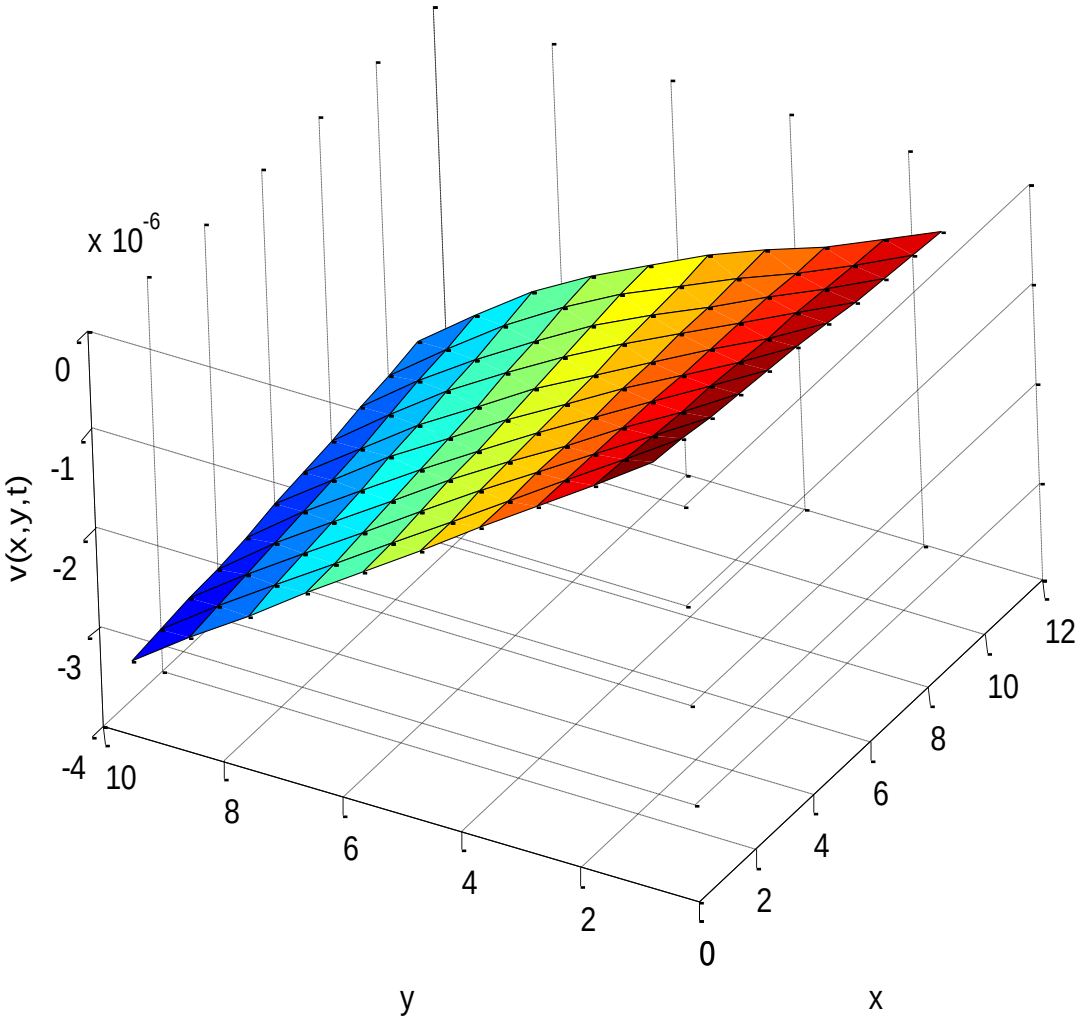

Figure 4.6: CN-LF Numerical Solution of *v*

The variation of the solution of $\boldsymbol{v}(\boldsymbol{x},\boldsymbol{y})$ shown in the Figure 4.6 is a smooth curve and does not portray any sudden change for various values of $\boldsymbol{x}$ and $\boldsymbol{y}$ for the CN-LF scheme. Thus the solutions from the developed scheme are consistent.

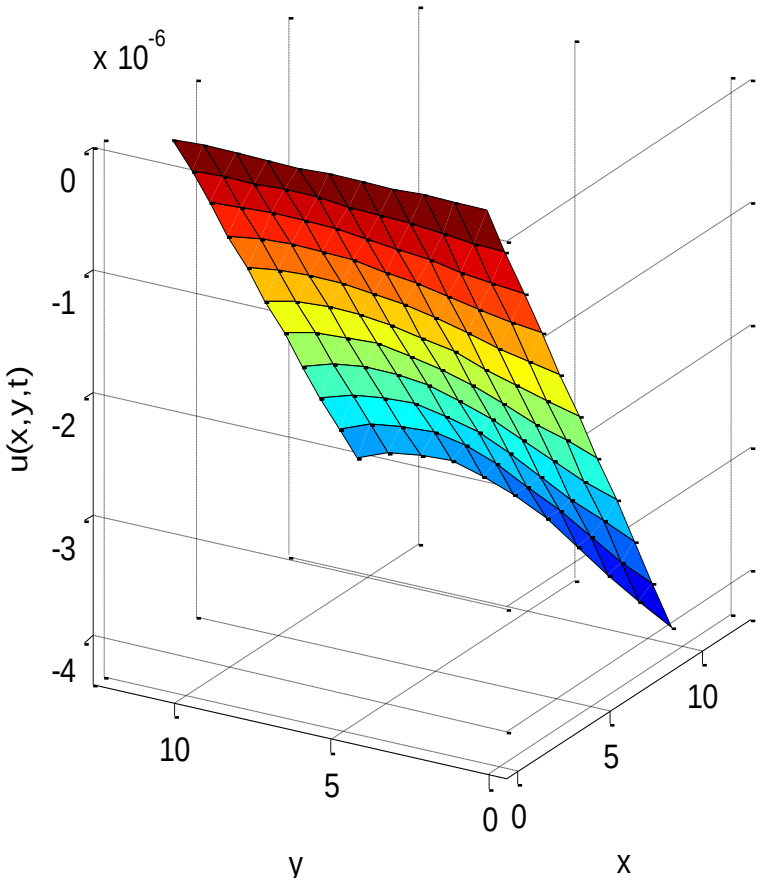

Figure 4.7: CN-DF Numerical Solution of *u*

The variation of the solution of $\boldsymbol{u(x,y)}$ shown in the Figure 4.7 is a smooth curve and does not portray any sudden change for various values of $\boldsymbol{x}$ and $\boldsymbol{y}$ for the CN-DF scheme. Thus the solutions from the developed scheme are consistent.

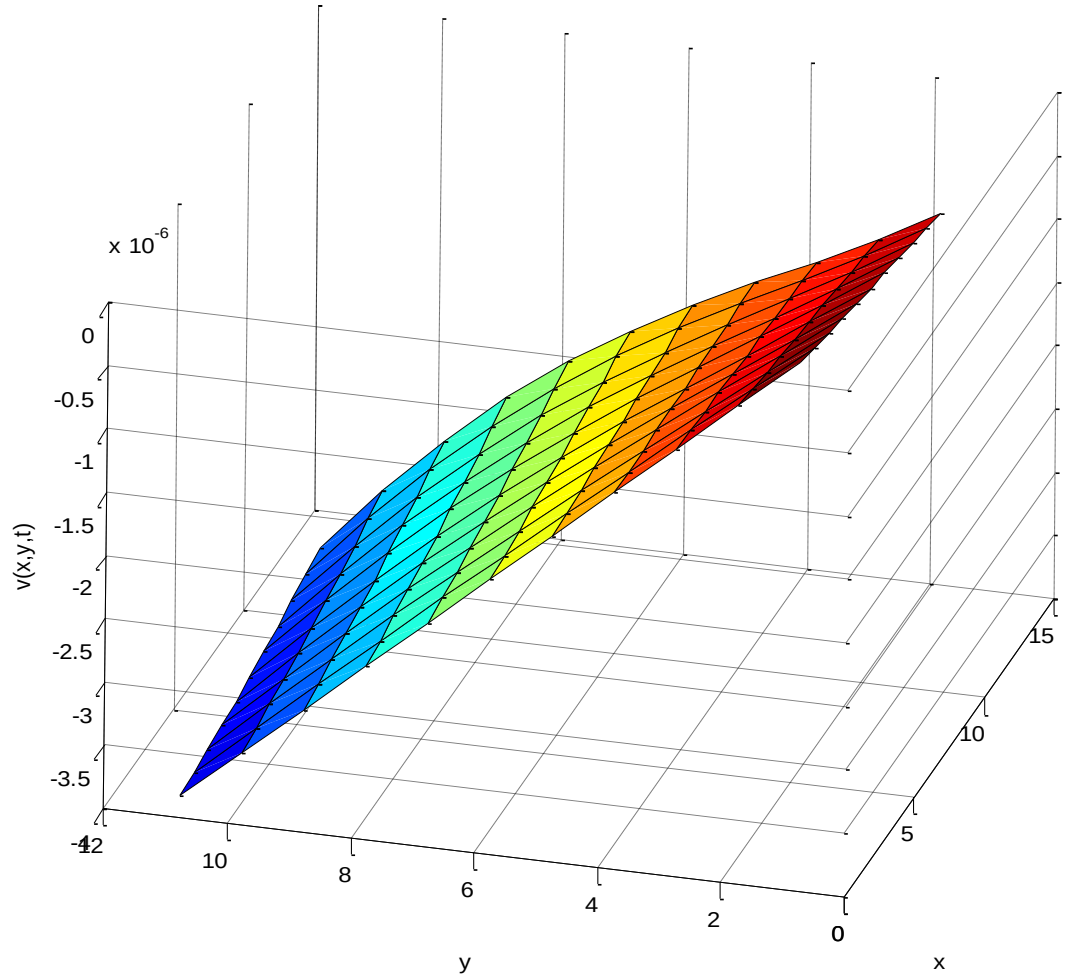

Figure 4.8: CN-DF Numerical Solution of *v*

The variation of the solution of $\boldsymbol{v(x,y)}$ shown in the Figure 4.8 is a smooth curve and does not portray any sudden change for various values of $\boldsymbol{x}$ and $\boldsymbol{y}$ for the CN-DF scheme. Thus the solutions from the developed scheme are consistent.

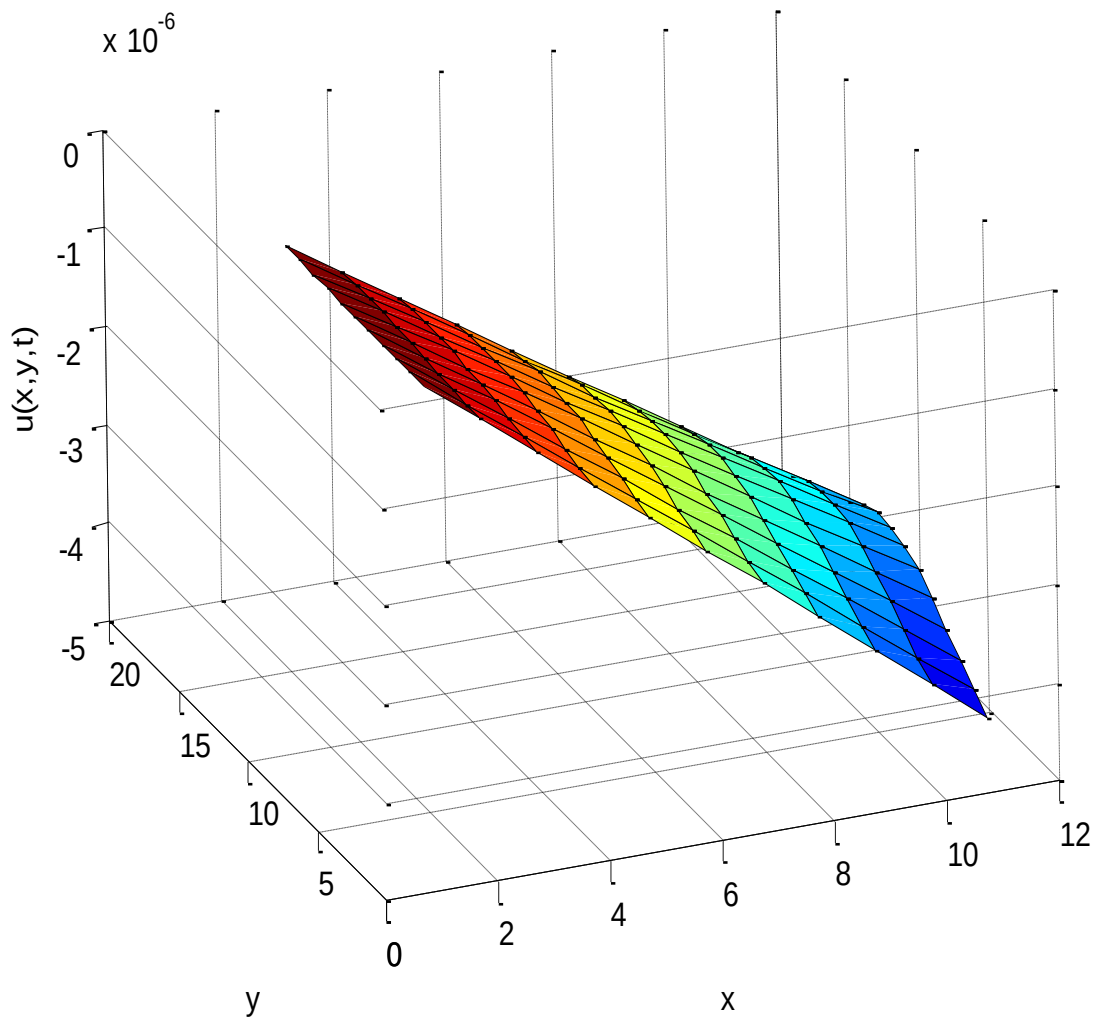

Figure 4.9: CN-DF-LF Numerical Solution of *u*

The variation of the solution of $\boldsymbol{u(x,y)}$ shown in the Figure 4.9 is a smooth curve and does not portray any sudden change for various values of $\boldsymbol{x}$ and $\boldsymbol{y}$ for the CN-DF-LF scheme. Thus the solutions from the developed scheme are consistent.

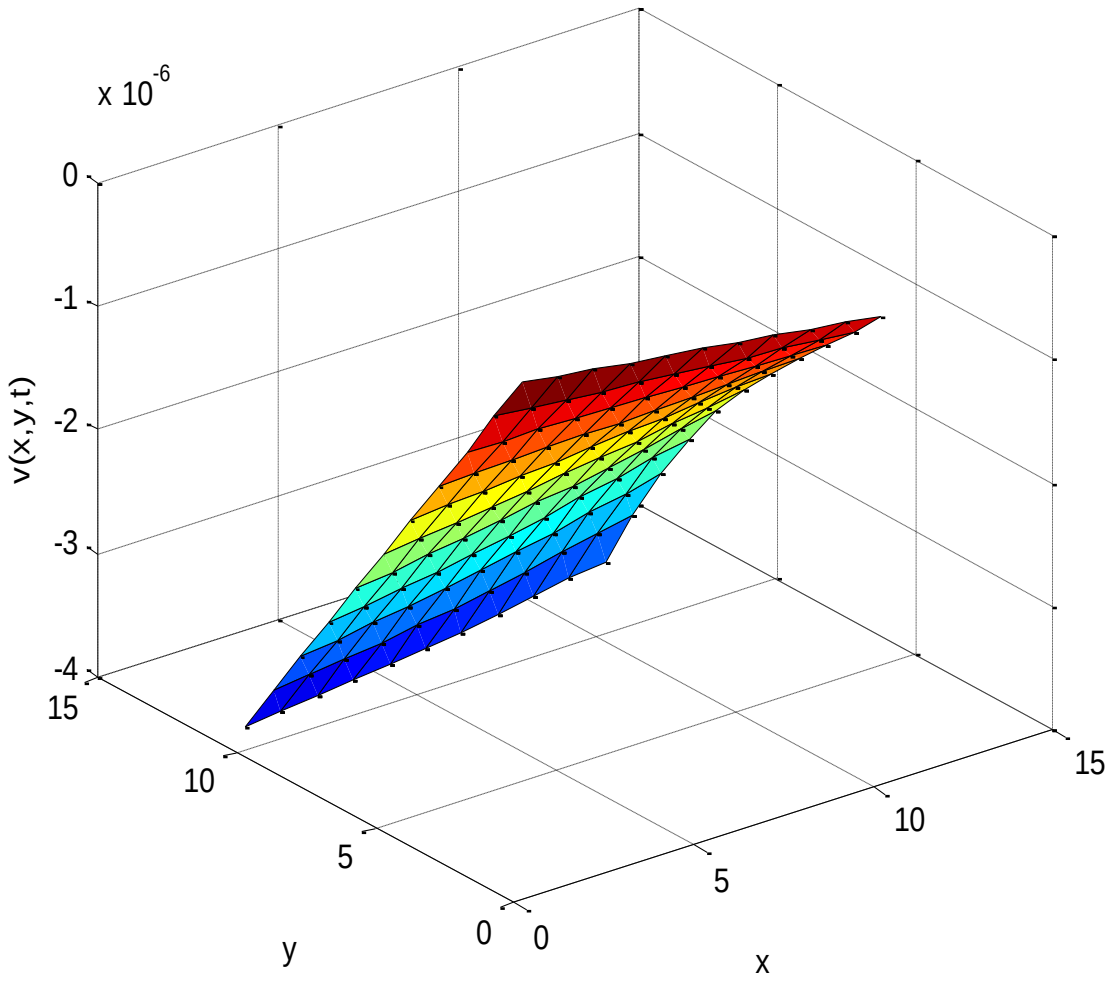

Figure 4.10: CN-DF-LF Numerical Solution of *v*

The variation of the solution of $\boldsymbol{v(x,y)}$ shown in the Figure 4.10 is a smooth curve and does not portray any sudden change for various values of $\boldsymbol{x}$ and $\boldsymbol{y}$ for the CN-DF-LF scheme. Thus the solutions from the developed scheme are consistent.

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

5.1 Introduction

In this chapter the conclusions are discussed and suggestions on some recommendations for further study are made.

5.2 Conclusion

The study has successfully developed the Crank-Nicholson (CN), Hybrid Crank-Nicholson-Du Fort & Frankel (CN-DF) and Hybrid Crank-Nicholson-Du Fort & Frankel-Lax Fredrich's (CN-DF-LF) scheme from Operator Splitting.

The developed schemes have proved to be stable because the 3-D graphs drawn proved to be consistent and convergence was also tested. The schemes proved to satisfy the consistence, convergence and stability theory. The CN-DF-LF is most accurate than the pure CN, CN-DF and CN-LF scheme. The decrease in the absolute error verifies the consistency, convergence and hence stability of the solutions from schemes developed.

The developed schemes can be applied to 2-D non-linear parabolic equations similar to the Burgers' equation.

5.3 Recommendation

Further research can be done on the following areas:

i) The effect of varying the Reynolds number on the solution of 2-D Burgers equation by the Hybrid Finite difference schemes.

ii) Numerical solution of higher level non-linear parabolic equations similar to Burger's equation.

iii) Use of operator splitting method to solve other non-linear equations like the Korteweg and de Vries (KdV)

REFERENCES

Ali, A. (2009). *Mesh free collocation method for numerical solution of initial-boundary-value problemsusing radial basis functions.* Pakistan: GhulamIshaq Khan Institute of Engineering Sciences and Technology.

Al-Saif , A., Mohammed , J., & Al-Kanani. (2012). Alternating Direction Implicit Formulation of the Differential Quadrature Method for Solving Burger Equations. *International Journal of Modern Mathematical Sciences, 3*(1), 1-11.

Ames, W. (1992). *Numerical Methods for Partial Differential Equations.* New York: Academic Press Inc.

Aminikhah, H., & Moradia, S. (2014). Numerical solution for the systems of variable-coefficient coupled Burgers' equation by two dimensional Legendre wavelets method. *Applications and Applied Mathematics: An International Journal (AAM), 9*(1), 342-361.

Amruta , D., & Vikas , P. (2014). A Novel Approach for Solving Burger's Equation. *Applications and Applied Mathematics: An International Journal (AAM), 9*(2), 541-552.

Ashyralyev, A., Erdogan, A. S., & Arslan, N. (2010). On the modified Crank–Nicholson difference schemes for parabolic equation with non-smooth data arising in biomechanics. *Int. J. Numer. Meth. Biomed. Engng.*, 501-510.

Baruch, C. (1995). Stepwise Stability for the heat equation with a non local constraints. *SIAM Journal of Numerical Analysis, 32*(2), 571-593.

Basto , M., Semiao, V., & Calheiros, F. (2009). Dynamics and synchronization of numerical solutions of the Burgers equation. *J. Comput. Appl. Math., 231*, 793-803.

Bateman, H. (1915). Some recent researches on the motion of fluids. *Monthly Weather Rec., 43*, 163-170.

Beauchamp, C., & Arminjon, P. (1979). Numerical solution of Burgers' equations in two-space dimensions. *Comput. Meth. Appl. Mech. Eng., 19*(3), 351-365.

Blanes, S., Casasy, F., & Muruaz, A. (2006). *Symplectic splitting operator methods for the time-dependent Schrodinger equation.* Spain.

Borah , A., Singh, P., & Goswami, P. (2012). A Current Perspectives of Corrected Operator Splitting (OS) for Systems. *Mathematical Theory and Modeling, 2*(8), 72-78.

Boyd, S., Parikh, N., & Chu, E. (2013). *Monotone Operator Splitting Methods (Lecture notes).* Stanford University.

Burgers, J. M. (1948). A Mathematical Model illustrating the theory of turbulence. *Advances in Applied Mechanics, 1*, 171-199.

Chang, M. (1991). Improved alternating-direction implicit method for solving transient three-dimensional heat diffusion problems, Numerical Heat Transfer. *19*, 69-84.

Chapra, S., & Canale, R. (1998). *Numerical methods for Engineers.* WCB/ McGraw-hill.

Chen, L. (2013). *Finite Difference Methods.* Irvine: University of California.

Chertock, A., Kurganov, A., & Petrova, G. (2004). *Fast Explicit Operator Splitting Method Application to the Polymer System.* North Carolina State University.

Cole, J. (1951). On a quasi-linear parabolic equation occurring in aerodynamics. *Quarterly Reports of Applied Mathematics, 9*, 225-236.

Delgado, P. (2013). *Operator Splitting Methods for PDE's.* University of Texas at El Paso.

Douglas, J. (1962). Alternating direction methods for three space variables. *Numerische Mathematik, 4,* 41-63.

Du Fort , E., & Frankel, S. (1953). Conditions in the Numerical Treatment of Parabolic Differential Equations. *Mathematical Tables and Other Aids to Computation, 7,* 135-152.

Espen, B. N. (2011). *On Operator Splitting for the Viscous Burgers' and the Korteweg-de Vries Equation.* Thesis, Norwegian University of Science & Technology, Department of Mathematical Sciences, Norway.

Evje , S., & Hvistendahl , K. (1999). Viscous Splitting Approximation of mixed Hyperbolic-parabolic Convention-Diffusion equation. *Numer. Math., 83,* 107-137.

Geiser, J. (2001). *Numerical Analysis and Scientific Computation: Iterative Splitting Methods for Differential Equations.* Boca Ranton, Florida: CRC Press/ Taylor and Francis Group: A chapman & Hall Book.

Geiser, J., & Noack, L. (2008). *Operator-Splitting Methods Respecting Eigenvalue Problems for Nonlinear Equations and Applications for Burgers Equations.* Berlin.

Gottlieb, D., & Gustafsson, B. (1976, March). Generalized Du Fort-Frankel Methods for Parabolic Initial-Boundary Value Problems. *SIAM Journal on Numerical Analysis, 13*(1), 129-144.

Gurevich, S. (2008). *Numerical Methods for complex systems.* Institute for Theoretical Physics, University of Münster.

Hairer, M., & Voss , J. (2010). *Approximations to the Stochastic Burgers Equation.*

Han, H.-d., Wu, X.-n., & Xu, Z.-l. (2006). Artificial Boundary Method for Burgers' Equation Using Nonlinear Boundary Conditions. *Journal of Computational Mathematics, 24*(3), 295–304.

Hockbruck, M., & Osterman, A. (2005). *Time Integration: Splitting Methods* . Helsinki.: CPIP.

Holden, H., Hvistendahl , K., & Lie , K.-A. (2000). Operator splitting methods for degenerate convection–diffusion equations II: numerical examples with emphasis on reservoir simulation and sedimentation. *Computational Geosciences, 4*, 287-322.

Holden, H., Lubich, C., & Risebro, N. (2011). *Operator Splitting for Partial Differential Equations with Burgers Non-linearity*. math. AP.

Hongqing, Z., Huazhong, S., & Meiyu, D. (2010). Numerical solutions of two-dimensional Burgers' equations by discrete Adomian decomposition method. *Computers and Mathematics with Applications, 60*, 840-848.

Hopf, E. (1950). The partial differential equation u_t+uu_x=u_xx. *Communications on Pure and Applied Mathematics, 3*, 201-230.

Idris, D., & Ali, S. (2007). Numerical solution of the Burgers' equation over geometrically graded mesh. *Kybernetes, 36*(5/6), 721-735.

Iltaf , H., Safyan, M., & Arshed, A. (2013). A Numerical Meshless Technique for the Solution of the two Dimensional Burger's Equation Using Collocation Method. *World Applied Sciences Journal, 23*(12), 29-40.

Istvan, F. (2003). Lecture Notes on splitting Methods. *SIAM journal of Numerical Analysis, 33*, 48-57.

Jain, M. K. (2004). *Numerical methods for scientist and engineering computation.* Wiley eastern limited.

Jiang , Z., & Wang, R. (2010). An Improved Numerical Solution of Burgers' Equation by Cubic B-spline Quasi-interpolation. *Journal of Information & Computational Science, 7*(5), 1013-1021.

Koross, A., Chepkwony, S., Oduor, M., & Omolo, O. (2009). Implicit Hybrid Finite Difference Methods Arising from Operator Splitting for solving 1-D Heat Equation. *Journal of Mathematical Sciences, 20*(1), 75-82.

Krishnan, D., Lin, P., & Tai, X. (2006). An Efficient Operator-SplittingMethod for Noise. *Communications in Computational Physics, 1*(5), 847-858.

Kutluay, S., & Yagmurlu, N. (2012). The Modified Bi-quintic B-Splines for Solving the Two-Dimensional Unsteady Burgers' Equation. *European International Journal of Science and Technology, 1*(2), 23-39.

Kweyu , M. C., Manyonge, W. A., Koross A. , A., & Ssema. (2012). Numerical Solutions of the Burgers' System in Two Dimensions under Varied Initial and Boundary Conditions. *Applied Mathematical Sciences, 6*(113), 5603-5615.

Kweyu, C., Nyamai, B., & Wahome, J. (2014). Hybrid Crank-Nicolson-Du Fort and Frankel (CN-DF) Scheme for the Numerical Solution of the 2-D Coupled Burgers' System. *Applied Mathematical Sciences, HIKARI Ltd, www.m-hikari.com, 8*(48), 2353-2361.

Le Veque, R. J., & Oliger, J. (1983). Numerical methods based on Additive splitting of hyperbolic Partial Differential Equations. *Mathematics for computation, 40*(16), 469-497.

Levandosky, J. (2001). *Partial differential Equations of Applied Mathematics (Lecture Notes).* Stanford University.

Luo , X. (1996). Operator-Splitting Computation of Turbulent Flow in an Axisymmetric 180° Narrowing end Using Several k-ε Models and Wall Functions. *International Journal for Numerical Methods in Fluids, 22*, 1189-1205.

Marchuk, G. (1968). Some applications of splitting-up methods to the solution of problems in mathematical physics. *Aplikace Matematiky 13, 13*, 103-132.

Mitchel , A. R., & Grffiths , D. F. (1980). *The Finite Difference Method in Partial Differential Equations.* John Wiley & sons.

Morton, K., & Mayers, D. (2005). .*Numerical Solution of Partial Differential Equations, An Introduction.* Cambridge University Press.

Nurcan, G., & Gamze, T. (2011). Iterative operator splitting method for capillary formation model in tumor angiogenesis problem: Analysis and application. *International Journal of Numerical Methods of Biomedical Engineering, 27*, 1740–175.

Peaceman, D. W., & Rachford, H. J. (1955). The numerical solution of parabolic and elliptic differential equations. *SIAM Journal, 3*, 28-41.

Qimiao, L., & Onyx , W. (1998). An Efficeint Operator Splitting Scheme for Three-Dimensional Hydrodynamic Computations. *International journal for numerical methods in fluids, 26*, 771-789.

Rahman, M. (1998). . *Partial Differencial Equations.* Southamptom Boston.: Computational Mechanics publication.

Rao, K. (2005). *Numerical Methods for Scientists and Engineers* (2nd ed.). New Delhi: Prentice Hall of India private Limited.

Rao, S., & Yadav, M. (2010). On the Solution of a Nonhomogeneous Burgers Equation. *International Journal of Nonlinear Science, 10*(2), 141-145.

Rashidi , M., & Erfani , E. (2009). New analytical method for solving Burgers' and nonlinear heat transfer equations and comparison with HAM. *Comput. PhysCommun, 180*, 1539-1544.

Shafiqul, I., Saiduzzaman, Sobuj, K., Shifat, A., & Sazzad, L. (2014). A nUmerical Experiment on Burger's Equation. *International Journal of Scientific & Engineering Research, 5*(4), 491-499.

Srivastava , V., Tamsir , M., Bhardwaj , U., & Sanyasir. (2011). Crank-Nicolson scheme for numerical solutions of two-dimensional coupled Burgers' equations. *International Journal of Scientific & Engineering Research, 2*(5), 1-7.

Srivastava, V., Awasthi, M., & Tamsir, M. (2013). A fully implicit Finite-difference solution to one dimensional Coupled Nonlinear Burgers' equations. *International Scholarly and Scientific Research & Innovation, 7*(4), 417-422.

Strang, G. (1963). Accurate partial difference methods I: Linear Cauchy problems,. *Arch. Rational Mech. Anal., 12*, 392–402.

Strikwerda, J. (1989). *Finite Difference Schemes and Partial Differential Equations.* Chapman & Hall.

Tadjeran, C. (2007). Stability analysis of the Crank–Nicholson method for variable coefficient diffusion equation. *Commun. Numer. Meth. Engng, 23*, 29-34.

Teukolsky , S. (1999). *On the Stability of the Iterated Crank-Nicholson Method in Numerical Relativity.* NewYork: Newman Laboratory, Cornell University, Ithaca.

Trotter, H. (1959). On the product of semi-groups of operators. *Proceedings of The American Mathematical Society, 10*(4), 545-551.

Vineet , K. S., Mohammad , T., Utkarsh , B., & Sanyasira, Y. (2011). Crank-Nicolson Scheme for Numerical Solution of Two-Dimensional Coupled Burgers' Equations. *International Journal of Scientific and Engineering Research,, 2*, 2229-5518.

Weinan, E. (1992). Convergence of Spectral Methods for Burgers' Equation. *Siam Journal of Numerical Analysis, 29*(6), 1520-1541.

Wille, S. (1998). Nodal Operator Splitting Adaptive Finite Element Algorithms for the Navier-Stokes Equations. *International journal for numerical Methods in fluids, 26*, 959-975.

Yesim , C., & Gamze , T. (2015, April 1). CMMSE-Convergence Analysis for Operator Splitting Methods with Application to Burgers-Huxley Equation. *Applied Mathematics and Information Sciences: An International Journal, 9*(2), 407-414.

Yesim, Y. (2010). *Operator Splitting Methods for Differential Equations.* Thesis, Graduate School of Engineering and Sciences of Izmir Institute of Technology, Department of Mathematics.

Zheng, B. (2010). Traveling wave solutions for the (2+1) dimensional Boussinesq equation and the two-dimensional Burgers equation by (G'/G)-expansion method. *WSEAS Transactions on Computers, 9*(6), 614-623.