To the Graduate Council:

I am submitting herewith a thesis written by Yanping Shi entitled "Studies on the generalized Roscoe procedure for incompressible flow" I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in aerospace engineering.

Robert L. Roach

Dr. Robert L. Roach, Major Professor

We have read this thesis
and recommend its acceptance:

Accepted for the Council:

Associate Vice Chancellor
and Dean of the Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at The University of Tennessee, Knoxville, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of the source is made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature _____

Date April 28 95

STUDIES ON THE GENERALIZED ROSCOE PROCEDURE FOR INCOMPRESSIBLE FLOW

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Yanping Shi

May 1995

DEDICATION

This thesis is dedicated to my parents:

Shulin Shi

and

Quiying Sun

without whose love and support, I would not be where I am today.

ACKNOWLEDGMENTS

I would like to take this opportunity to thank my advisor, Dr. Robert Roach, for his guidance and support during the course of my graduate education here at UTSI. His unending help and contributions to this work are greatly appreciated.

I would also like to thank the other members of my thesis committee, Dr. Trevor Moulden and Dr. Gary Flandro.

Special thanks goes to Dan Thomas, Robert Maddock and especially Allen McCarley who were very helpful with the preparation of this thesis. A very special thanks goes to Dr. Trevor Moulden who took time out of his busy schedule to listen to my ideas and discuss the details with me. Special appreciation goes to the University of Tennessee Space Institute library, headed by Mary Lo, for helping with the vast literature search that was required for this research.

I would like to express my sincere thanks to all of my friends, family and especially my dear husband, Zhida Ma, who supported me and provided constant encouragement throughout my master's studies here at UTSI. A very special thanks go to my lovely little daughter, Vivian Yan Ma, for her sacrifices of time with her mother while this work has been done.

ABSTRACT

A generalization of the Roscoe discretization for the unsteady, incompressible Navier-Stokes equations is studied in the present work. The distinction of this generalization lies in the systematic approach adopted for the discretization. The resulting discretized model generated from a linear differential system, which represents an approximation for the given nonlinear problem. A phase space study on one of the test problems, the nonlinear Burgers equation suggests that this approximate system of linear differential equations preserves the system stability of the original nonlinear equation. One of new features of the generalized method is that *unique* interpolation functions containing some of the characteristics and properties of the original system (in a approximate sense) are generated for use in the discretization. These unique interpolation functions are found to be a linear or quadratic function plus a exponential term. This is in contrast to most finite difference or finite volume methods where the interpolation functions are usually some low order polynomials. The interpolation functions are obtained by integrating locally linearized forms of the governing equations as a system of ordinary differential equations in one direction at a time. There is thus a unique function for each dependent variable in each coordinate direction. Since the equations are integrated as a system, interactions between the equations are retained in the resulting interpolation functions. Corresponding to the two constructions of the interpolation functions, two numerical schemes, a linear exponential spline and a quadratic exponential spline scheme, have been developed and applied to the steady state nonlinear Burgers equation and to the unsteady, incompressible Navier-Stokes equations for the Lamb vortex problem. The numerical results are compared with those from the original Roscoe scheme, second-order central differencing, a cubic spline method and a fourth-order compact method. A stability and accuracy analysis is also given with conclusions which fit well with the computational results. The two exponential spline schemes derived from the generalized Roscoe procedure are found to be unconditionally stable with respect to cell Reynolds number. Both of the numerical solutions and the analytical studies show that these exponential spline solving procedures, as developed, have second order accuracy with respect to the spatial grid size.

TABLE OF CONTENTS

Chapter		Page
1.	**Introduction**	1
2.	**Derivation of the Generalized Roscoe Discretizaion**	5
	2.1 Introduction of the Roscoe Scheme	5
	2.2 Generalization of the Roscoe Scheme	11
	2.3 Phase Space Study for the Roscoe Scheme	24
3.	**The Steady State Nonlinear Burgers Equation**	31
	3.1 The Linear Exponential Spline Interpolation	32
	3.2 The Quadratic Exponential Spline Interpolation	35
4.	**The Lamb Vortex Problem**	41
	4.1 Description of the Problem	41
	4.2 Numerical Procedure for the Linear Exponential Spline Method	42
	4.3 Numerical Procedure for the Quadratic Exponential Spline Method	44
	4.4 Numerical Procedure for the Cubic Spline Method	45
5.	**Stability and Accuracy Study**	46
	5.1 Accuracy Study	49
	5.2 Stability Study	64
	5.3 Analysis of the Numerical Solutions	71
6.	**Conclusions and Recommendations**	75
List of Reference		77
Vita		81

LIST OF FIGURES

Figure		Page
Figure 2.2.1:	Definition of the Linear Exponential Spline	23
Figure 2.2.2	Definition of the Quadratic Exponential Spline	24
Figure 2.3.1	Phase Space for the Nonlinear Burgers Equation	26
Figure 2.3.2	Phase Space for the Linearized Burgers Equation	27
Figure 5.1.1	Numerical Solutions for Burgers Equation ($Re_{cell} = 0.625$)	47
Figure 5.1.2	Numerical Solutions for Burgers Equation ($Re_{cell} = 1.25$)	47
Figure 5.1.3	Numerical Solutions for Burgers Equation ($Re_{cell} = 2.5$)	48
Figure 5.1.4	Numerical Solutions for Burgers Equation ($Re_{cell} = 5.0$)	48
Figure 5.1.5	Numerical Solutions for Burgers Equation ($Re_{cell} = 50$)	49
Figure 5.1.6	Numerical Solutions for Burgers Equation ($Re_{cell} = 500$)	49
Figure 5.1.7	Phase Space Solutions for Burgers Equation ($Re_{cell} = 0.625$)	51
Figure 5.1.8	Phase Space Solutions for Burgers Equation ($Re_{cell} = 1.25$)	51
Figure 5.1.9	Phase Space Solutions for Burgers Equation ($Re_{cell} = 2.5$)	52
Figure 5.1.10	Phase Space Solutions for Burgers Equation ($Re_{cell} = 5.0$)	52
Figure 5.1.11	Accuracy Comparison of the Numerical Solutions of Burgers Equation	53
Figure 5.1.12	Numerical Solutions for the Lamb Vortex on the 100x100 grid at t = 12.28 sec	55
Figure 5.1.13	Numerical Solutions for the Lamb Vortex on the 80x80 grid at t = 12.28 sec	55
Figure 5.1.14	Numerical Solutions for the Lamb Vortex on the 60x60 grid at t = 12.28 sec	56
Figure 5.1.15	Accuracy Comparison of the Numerical Solutions to the Lamb Vortex	56
Figure 5.1.16	Interior of the Linear Exponential Spline	58
Figure 5.2.1	Polar Plot of |G| for the Linear Exponential Spline Procedure at Large Range of Cell Reynolds Number	68
Figure 5.2.2	Polar Plot of |G| for the Linear Exponential Spline Procedure at Small Range of Cell Reynolds Number	69
Figure 5.2.3	Polar Plot of |G| for the Linear Exponential Spline Procedure for variable cell Reynolds number.	23

TABLE OF SYMBOLS

a	coefficient of differential equation, or, the left boundary of a defined discretized domain
a_1, a_2, a_3	coefficients in the von Neumann Stability study
a_m, a_i, a_p	coefficients in the equation for the equivalent solving system for the linear exponential spline procedure
A_1, A_2, A_3	coefficients in the von Neumann Stability study
A	coefficient in discretized expression of velocity derivatives
a_i	coefficient of the spline interpolation function
a_{x_u}	coefficient of x direction spline interpolation function for u(x,y)
a_{x_v}	coefficient of x direction spline interpolation function for v(x,y)
a_{x_p}	coefficient of x direction spline interpolation function for p(x,y)
a_{y_u}	coefficient of y direction spline interpolation function for u(x,y)
a_{y_v}	coefficient of y direction spline interpolation function for v(x,y)
a_{y_p}	coefficient of y direction spline interpolation function for p(x,y)
b	coefficient of differential equation, or, the right boundary of a defined discretized domain
b_1, b_2, b_3	coefficients in the von Neumann Stability study
b_m, b_i, b_p	coefficients in the equation for the equivalent solving system for the linear exponential spline procedure
B	coefficient in discretized expression of velocity derivatives
B_1, B_2, B_3	coefficients in the von Neumann Stability study
b_i	coefficient of the spline interpolation function
b_{x_u}	coefficient of x direction spline interpolation function for u(x,y)
b_{x_v}	coefficient of x direction spline interpolation function for v(x,y)
b_{x_p}	coefficient of x direction spline interpolation function for p(x,y)
b_{y_u}	coefficient of y direction spline interpolation function for u(x,y)
b_{y_v}	coefficient of y direction spline interpolation function for v(x,y)
b_{y_p}	coefficient of y direction spline interpolation function for p(x,y)
C	coefficient in discretized expression of velocity derivatives
c_i	coefficient of the spline interpolation function
c_{x_u}	coefficient of x direction spline interpolation function for u(x,y)
c_{x_v}	coefficient of x direction spline interpolation function for v(x,y)
c_{x_p}	coefficient of x direction spline interpolation function for p(x,y)

c_{y_u}	coefficient of y direction spline interpolation function for u(x,y)
c_{y_v}	coefficient of y direction spline interpolation function for v(x,y)
c_{y_p}	coefficient of y direction spline interpolation function for p(x,y)
D	coefficient in discretized expression of velocity derivatives
d_i	coefficient of the spline interpolation function
dx	grid size
d_{x_u}	coefficient of x direction spline interpolation function for u(x,y)
d_{x_v}	coefficient of x direction spline interpolation function for v(x,y)
d_{x_p}	coefficient of x direction spline interpolation function for p(x,y)
d_{y_u}	coefficient of y direction spline interpolation function for u(x,y)
d_{y_v}	coefficient of y direction spline interpolation function for v(x,y)
d_{y_p}	coefficient of y direction spline interpolation function for p(x,y)
$E_{v''}$	dissipative error
$E_{v'''}$	dispersion error
$E_{diffusion}$	diffusion error
f_1	nonlinear coefficient in differential equation
f_2	nonlinear coefficient in differential equation
f_3	forcing function in differential equation
G	amplification factor in von Neumann stability study
h	uniform spatial grid size
i	node point index
j	node point index
m	the left grid interval when used as a subscript
n	the total number of grid point, or, a certain time level
p	pressure, or, the right grid interval when used as a subscript
r	the linearized forcing function or radius in polar coordinates
$\vec{r}$	the approximated forcing function vector
$\vec{R}$	the approximated forcing function vector, or, right hand side vector
Re_{cell}	cell Reynolds number
$s_i(x)$	spline interpolate for interval i
S(x)	global spline interpolation function
s_i'	first derivative of spline interpolate on interval i
s_i''	second derivative of spline interpolate on interval i
$\vec{s}$	integration vector for $\vec{r}$

t	time
$\tilde{T}$	differencing operator in the original Roscoe method
u	velocity component in x direction
U	linearized velocity component for u
U_i	linearized velocity component U at node point i
v	velocity component in y direction
V	linearized velocity component for v
v_1	velocity in phase space study
v_2	velocity derivative in phase space study
V_i	linearized velocity component V at node point i
x	axis in Cartesian coordinate
y	axis in Cartesian coordinate
α	coefficient of approximated forcing function in differential equation
β	coefficient of approximated forcing function in differential equation
γ	coefficient of approximated forcing function in differential equation
δ	coefficient of approximated forcing function in differential equation
ε	viscosity, or, an arbitrary small number
λ	characteristic root for differential equation
λ_x	interpolation parameter in x direction
λ_y	interpolation parameter in x direction
σ	function of phase angle in the von Neumann stability study
ϕ	general form of the analytical solution to differential equation, or, phase angle in the von Neumann stability study
ψ	an arbitrary linear function of x and y
Δx	spatial step size
Δt	time step size

CHAPTER 1

INTRODUCTION

A general analytical solution to the governing equations in fluid dynamics, the Navier-Stokes equations, is not available due to nonlinearities in the system and to the lack of effective mathematical tools. It has not even been proven that a unique solution exists. As an alternative, numerical approaches are used with the help of modern computers (and, thereby, assume the existence of a unique solution).

The numerical approach, by concept, tries to simulate the partial differential equation (PDE) system with a discretized algebraic system and claims that under certain conditions, the numerical solution to this approximate system converges to that of the original differential equation. There are many ways to construct the algebraic equations which, presumably, simulate the behavior of partial differential equation systems. Unfortunately, some of these algebraic systems may have properties very different from those of the given PDEs. Thus, the computed solutions can vary significantly from the real solutions. In the sense of simulation accuracy, it is hoped that the algebraic system formulated during the discretization process retains the properties of the original system and lacks properties which do not belong to the original system. It is also hoped that the resulting numerical solving procedure has adequate numerical stability and accuracy.

The most frequently used discretization techniques include: the finite difference method, the finite element and finite volume methods and the spectral method. Each of these has its own advantages and disadvantages. For example, in the finite difference method, where the derivative terms in the differential equation are replaced directly by algebraic functions derived from Taylor expansions, the solutions are finite polynomials in the independent variables. The finite difference equation is thus easy to construct and it is also very easy to program. In cases where the PDE systems have solutions which are close to low order polynomials or there are no singular points or lines in the flow fields, that is, no large gradients (large derivatives), this method gives quite good results. However, the shortcomings for this method is also very obvious. A brief discussion about this is given here.

The finite difference method is based upon the properties of Taylor expansions:

$$u(x+\Delta x) = u(x) + \Delta x u_x(x) + \frac{\Delta x^2}{2!} u_{xx}(x) + \frac{\Delta x^3}{3!} u_{xxx}(x) + \cdots \qquad (1.1)$$

By expanding the u(x) around the node point i with equation (1.1), an algebraic expression for $u_x(i)$ or $u_{xx}(i)$ can be obtained in terms of nearby node point values. In this process, only a finite number of terms in the infinite Taylor expansion are kept and an infinite number of high order terms are truncated. After truncating the high order terms, the finite difference method essentially approximates the real solution to a certain order polynomial. In the Weierstrass Approximation Theorem [1], it is specified that a polynomial exists that approximates (in the sense of uniform convergence) the solution (which is assumed to be smooth) but this polynomial must, in most cases, be of very high order.

It is known from analysis [2] that even and odd order derivative terms represent dissipation and dispersion effects respectively. So the physical properties contained in the high order terms are lost when these terms are truncated in the finite difference discretization. Thus it can be expected that the finite difference system behaves differently from the given PDE system. This difference becomes significant when the truncated terms are large in their magnitude, that is, when either the grid size is large or the derivatives themselves are large so that the truncation errors are no longer negligible. The oscillatory type of behavior sometimes observed from the central difference method is a good example of this. El-Mistikawy and Werle [3] showed that this oscillation is a property of the difference equation system rather than that of the given differential equation system. It may be concluded that by explicitly truncating the high order terms in the discretization, the system properties are inevitably changed. In some situations this property change causes severe problems. For example, when the central difference method is applied to solve for a flow with large cell Reynolds number, either the system is no longer stable or the oscillations are so strong that the numerical solution is not acceptable. To eliminate the oscillations or to stabilize the solution, some extra terms, called *artificial* viscosity terms, have been added to the difference equations. By adding the artificial viscosity terms, the algebraic structure of the algorithm changes and the solution may be forced to converge towards the intended values.

The other drawback of the finite difference method arises from the term by term discretization so that the interactions among the derivative terms in the original system are neglected. In multidimensional problems, the interpolation functions for the same dependent variable may be different from one direction to another direction and, again, the interactions among the directions are overlooked.

Essentially, the Taylor series based discretization methods approximate the real solutions with polynomials of some order. In principle, the higher the order of the polynomial, the more accurate is the solution. However, the approximate polynomials cannot be taken to as high an order as needed. Further, low order polynomials require an extremely large number of grid point for accuracy as well as the artificial viscosity terms for stability. Sometimes these requirements are not realistic. It would be desirable to have a numerical scheme which preserves the properties from the original differential equation, has no numerical instability and gives the required accuracy without any artificial viscosity terms required.

The construction of an improved numerical scheme starts from studying the nature of the problem, the differential equation system. Usually, the difficulty in solving a general nonlinear differential equation comes from the nonlinearity contained in the system. If the system is linear with constant coefficients an analytical solution is always available. In the numerical approach, a continuous mathematical model is modified into a discretized algebraic model. The information left after the discretization is a relation imposed on the node point values. Information about the system between the nodes, however, is lost in the discretization process. The obvious question becomes, then, is it possible, from the system simulation point of view, that information about the system between the node points be kept in the discretized model? The method studied here attempts to retain such information.

The present work is concerned with the relationship between the discrete local information contained in the construction of numerical algorithms and the global nonlinearity of the given system. The idea is similar to that of classical calculus wherein locally linearized information does, in the limit, $\Delta x \rightarrow 0$ generate information about nonlinear functions.

The specific technique adopted here is based on generalizing the work of Roscoe [4,5,6] who created finite difference expressions for the incompressible flow equations from the integration of convection-diffusion pairs. The generalized Roscoe discretization presented here starts from simulating the system of partial differential equations on a continuous domain with a set of locally linearized systems defined on discretized spatial intervals. Then, on each of these intervals, the corresponding linear differential equation is solved analytically. The resulting local analytical solutions are then taken as the interpolation functions which provide a spline basis for the calculation domain. The discretized algebraic equation contained in this global spline function represents the

numerical model used for the solving procedure. This procedure will be discussed in detail in the following chapters.

The technique of generating finite difference expressions by integrating linearized equations directly has been approached by several researchers. Allen and Southwell [7], (in 1955) encountered this technique in developing their relaxation method for a two dimensional viscous flow past a fixed cylinder. They found that, except within a very narrow range, the *inherent exponential in the solution* cannot be closely represented by a simple polynomial that would result from any finite difference representation. Spalding [8] suggested the same idea in 1972 but he did not carry it out because he thought the exponential terms that emerged in the procedure were too time-consuming to compute. Roscoe (1974) who first developed this idea into the so-called general UDR procedure [4] and applied it to several fluid flow problems [5], [6]. He achieved very good results when comparing with some Taylor expansion based schemes. Independently (work done in 1974 but published in 1977), Chien [9] published his generalized finite difference formulation which is identical to Roscoe's UDR procedure. El-Mistikawy (1977) and his colleagues further developed the Roscoe scheme into their exponential box scheme and applied it to solve for inviscid/viscous flows [3]. Later, Roach (1983) [10], Cahoon and Roach (1991, 1993) [11,12] made preliminary attempts at generalizing the procedure by adding the pressure gradient to the convection-diffusion pair integration. The results showed improvement over the original Roscoe procedure.

In the present research, Roscoe's approach (and that of Calhoon and Roach) has been extended to a more generalized level and the results have been applied to create two exponential spline schemes. The corresponding programs are developed and used to solve two test problems. Stability and accuracy studies have been carried out which fit quite well with the numerical results. So far, the results from the two exponential spline methods are promising when compared with some other classical schemes. It is found that the two exponential spline solving procedures are unconditionally stable with respect to cell Reynolds number and they both have second order accuracy with respect to spatial grid size.

CHAPTER 2

DERIVATION OF THE GENERALIZED ROSCOE DISCRETIZATION

In the first section of this chapter, the basic idea of the original Roscoe discretization is introduced in detail. The second section gives the derivation of the generalization on the original Roscoe method. A phase space study of the generalized Roscoe linearization is discussed in section 3 for the nonlinear Burgers equation.

2.1 Introduction of the Roscoe scheme

Roscoe [4] introduced a method that he referred as the Unified Difference Representation (UDR) which is introduced here. Unlike the traditional discretization methods, the discrete equations in the Roscoe method are derived directly from the analytical solution to a linear differential equation which is a simplification of the nonlinear one obtained by approximating the nonlinear coefficients. The following remarks are based closely in the work of Roscoe [4, 5, 6].

Consider a one-dimensional second-order nonlinear ordinary differential equation which, in a general form, is given by:

$$\frac{d^2\phi}{dx^2} + f_1(x,\phi)\frac{d\phi}{dx} + f_2(x,\phi)\phi = f_3(x,\phi); \tag{2.1.1}$$

with specified boundary conditions at $x = a$ and $x = b$. Notice that the nonlinear coefficients $f_1(x,\phi)$, and $f_2(x,\phi)$, the function of $f_3(x,\phi)$ are taken to be smooth functions of both x and ϕ. Consider solving this equation on a discrete domain:

$$a = x_1 < x_2 < \cdots < x_{i-1} < x_i < x_{i+1} \cdots < x_{n-1} < x_n = b \tag{2.1.2}$$

where, for simplicity (but not for necessity), a uniform grid size $h = x_{i+1} - x_i = x_i - x_{i-1}$ will be assumed for $i = 2,...,n-1$.

In a small region around the point i, $x_{i-1} \le x_i \le x_{i+1}$, $f_1(x,\phi)$ and $f_2(x,\phi)$ in equation (2.1.1) can be expanded with Taylor series:

$$f_1(x,\phi) = f_1(x_i,\phi_i) + \Delta x\left[\frac{\partial f_1}{\partial x}\right]_i + \Delta\phi\left[\frac{\partial f_1}{\partial \phi}\right]_i + \cdots$$
$$f_2(x,\phi) = f_2(x_i,\phi_i) + \Delta x\left[\frac{\partial f_2}{\partial x}\right]_i + \Delta\phi\left[\frac{\partial f_2}{\partial \phi}\right]_i + \cdots \qquad (2.1.3)$$

since they are assumed to be the smooth functions. By keeping only the first terms in equation (2.1.3), a linearized system corresponding to system (2.1.1) is obtained:

$$\frac{d^2\phi}{dx^2} + a_1\frac{d\phi}{dx} + a_2\phi = f_3 \qquad (2.1.4)$$
$$a_1 = f_1(x_i,\phi_i);\ a_2 = f_2(x_i,\phi_i)$$

Approximate boundary conditions for this system on a grid-cell sized subdomain will be the values at $x = x_{i-1}$ and $x = x_{i+1}$.

It is expected that on this small subdomain $x_{i-1} \le x_i \le x_{i+1}$, the system (2.1.4) will preserve equation type. This was shown by Roscoe [4] and is discussed below. At the same time, equation (2.14) preserves, in a linearized sense, the correct relation between ϕ and its derivative. This aspect of the Roscoe is discussed in section 2.3 on the phase space interpretation.

Through a proper solving process, the solution to equation (2.1.1) can be found from the local solution to equation (2.1.4) on every subdomain $x_{i-1} \le x_i \le x_{i+1}$, $i = 2,...,n-1$.

The solving process lies in constructing an algebraic equation based on the individual local analytical solution to equation (2.1.4). It is understood that the nonlinearity in equation (2.1.1) is brought back to the solution by using the *local* solutions on each of the grid interval which are derived from equation (2.1.4) with the *local* coefficient values. Thus, in the discretization process, from the solution point of view, there are no high order terms truncated explicitly as in the finite differencing. The accuracy of the numerical solution is influenced by the accuracy of the approximation of the nonlinear coefficients. Intuitively, the nonlinearity is resolved better with a small grid size. When the grid size tends to zero, the equation (2.1.4) tends to equation (2.1.1), hence, the solution to equation (2.1.4) tends to that of equation (2.1.1); essentially in this limit. When the grid size is large, the accuracy in the coefficient approximation decreases so that more of the nonlinearity tends to be destroyed. Thus, the accuracy of the solution decreases.

From the standard theory of ordinary differential equations, it is known that the general solution to equation (2.1.4) is:

$$\phi_i(x) = A_{1_i} e^{\lambda_{1_i} x} + A_{2_i} e^{\lambda_{2_i} x} + g_i(x) \qquad (2.1.5)$$

where i stands for the ith interval and A_1, A_2 are arbitrary constants which can be determined from specified boundary conditions. The constants λ_{1_i}, λ_{2_i} are the roots of the characteristic function of system (2.1.4) which depends on the homogenous part of equation (2.1.4) (that is, on the linearized functions f_1 and f_2) and may be complex. g(x) is a particular solution to equation (2.1.4) depending directly upon the forcing function f_3. Since equation (2.1.5) will be used to create the finite difference representation of equation (2.1.1), it represents the interpolating function for ϕ.

The stability of the system (2.1.4) at point 'i' is determined by the sign of the real part of λ_{1_i} and λ_{2_i}. Thus the solution in equation (2.1.5) contains this important stability information through these exponential terms.

To construct the discretized expression, the homogenous equation is first considered. When f_3=0, equation (2.1.5) becomes:

$$\phi(x) = A_{1_i} e^{\lambda_{1_i} x} + A_{2_i} e^{\lambda_{2_i} x} \qquad (2.1.5a)$$

On the subdomain $x_{i-1} \le x_i \le x_{i+1}$, $\phi(x)$ is evaluated with equation (2.1.5a) at the nodal points x_{i-1}, x_i and x_{i+1}. For the case of a uniform grid, a finite difference equation may be written (in the notation of reference [4]) as:

$$c_{1_i}\phi_{i+1} - c_{2_i}\phi_i + c_{3_i}\phi_{i-1} = 0 \qquad (2.1.6)$$

where:

$$c_{1_i} = 1$$

$$c_{2_i} = (e^{\lambda_{1_i} h} + e^{\lambda_{2_i} h})$$

$$c_{3_i} = e^{(\lambda_{1_i} + \lambda_{2_i})h}$$

In the case of periodic solutions (that is when $a_1^2 < 4a_2$ and the characteristic equation has complex roots):

$$
\begin{aligned}
c_{1_i} &= 1 \\
c_{2_i} &= e^{-a_1 h/2}\cos\left(\frac{\sqrt{4a_{2_i} - a_{1_i}^2}}{2} h\right) \\
c_{3_i} &= e^{-a_1 h}
\end{aligned}
\tag{2.1.7}
$$

By substitution of equation (2.1.6) into equation (2.1.4), the general discretized expression for equation (2.1.1), with forcing function $f_3 \neq 0$, is obtained as follows:

$$
\begin{aligned}
&B_i(c_{1_i}\phi_{i+1} - c_{2_i}\phi_i + c_{3_i}\phi_{i-1}) = a_{3_i} \\
&B_i = \frac{a_{3_i}}{c_{1_i} - c_{2_i} + c_{3_i}} \\
&a_{3_i} = f_3(x_i, \phi_i)
\end{aligned}
\tag{2.1.8}
$$

If the boundary conditions for equation (2.1.8) are appropriately set, the equation can be solved numerically by an iterative process to determine the node point value ϕ_i.

In the process of deriving equation (2.1.8), f_1, f_2 and f_3 are approximated with their *local* values based upon a Taylor expansion. Therefore, the coefficients B, c_1, c_2 and c_3 are discrete functions of space which change from node point to node point. The idea here is that for a vanishingly small region around point i, the solution of the difference equation (2.1.8) (for arbitrary step length) contains the local physics of the original differential equation (2.1.1), at least in an approximate sense that the local values of a_{1_i} and a_{2_i} (obtained from f_1 and f_2) dictate the values of λ_{1_i} and λ_{2_i} which are included in equation (2.1.8).

The general difference equation (2.1.6) uses a three point stencil. When the method is applied to the equations of viscous flow, Roscoe found the UDR method has second order accuracy (but with significantly smaller absolute error when compared with second order Taylor expansion based schemes). However, the Roscoe method becomes first order in the case of inviscid flow [3]. Roscoe found, upon expanding the exponential terms in equation (2.1.6) in Taylor expansion (to the first three terms), the classical backward difference form of equation (2.1.1) is obtained. Alternatively, multiplying equation (2.1.6) through by $\exp(-(\lambda_1 + \lambda_2)h)$ and $\exp(-(\lambda_1 + \lambda_2)\frac{h}{2})$ in turn and expanding the resulting expressions to the first three terms yield the classical forward and central difference formulae, respectively. This indicates that the Roscoe scheme adjusts

itself implicitly to backward, central or forward difference forms depending purely on the characteristic values λ. The Roscoe discretization thus has the remarkable property of preserving the elliptic or hyperbolic nature of the original system: the equation type.

The same philosophy had been applied by Roscoe to multidimensional problems. In this case, the multidimensional system is first approximated by several one dimension subsystems. Each subsystem consists of the convection-diffusion pair in one coordinate direction which is very similar to equation (2.1.1) except that it is now contained in a multidimensional equation. The subsequent treatment of each individual subsystem is identical to that of a one dimension differential equation discussed above.

To demonstrate the procedure Roscoe [4] is again followed. A two dimensional second-order partial differential system which, after the appropriate simplification, can be written as a differential operator, T:

$$T\phi \equiv (a\frac{\partial^2\phi}{\partial x^2} + b\frac{\partial\phi}{\partial x}) + (c\frac{\partial^2\phi}{\partial y^2} + d\frac{\partial\phi}{\partial y}) + f\phi = 0 \qquad (2.1.9)$$

Equation (2.1.9) can be written as:

$$T_x\phi + T_y\phi + f\phi = 0 \qquad (2.1.10)$$

where T_x represents all the x-derivatives in the equations while T_y contains all the y-derivatives. T_x and T_y are then treated as ordinary differential operators in the manner of the previous discussion. Their finite difference representations $\tilde{T}_i \approx T_x$ and $\tilde{T}_j \approx T_y$ are then derived similarly to give:

$$\tilde{T}\phi_{i,j} \equiv k_0\tilde{T}_i\phi_{i,j} + k_1\tilde{T}_j\phi_{i,j} + f\phi_{i,j} = 0 \qquad (2.1.11)$$

where k_0 and k_1 are determined by the requiring that $\phi(x,y)$ be an arbitrary linear function of x and y, then, if

$$\tilde{T}\psi_{i,j} = T\psi(x,y) \qquad (2.1.12)$$

$\tilde{T}$ becomes:

$$\tilde{T}_i\phi_{i,j} = \phi_{i+1,j} - (1+e^{-b\Delta x/a})\phi_{i,j} + e^{-b\Delta x/a}\phi_{i-1,j} = 0$$
$$\tilde{T}_j\phi_{i,j} = \phi_{i,j+1} - (1+e^{-d\Delta y/c})\phi_{i,j} + e^{-d\Delta x/c}\phi_{i,j-1} = 0$$
$$k_0 = \frac{b}{\Delta x(1-e^{-b\Delta x/a})} \qquad (2.1.13)$$
$$k_1 = \frac{d}{\Delta y(1-e^{-d\Delta y/c})}$$

It follows directly that

$$\begin{aligned} \text{sign}(k_0) &= \text{sign}(a) \\ \text{sign}(k_1) &= \text{sign}(c) \end{aligned} \qquad (2.1.14)$$

Roscoe [4] found two important properties of $\tilde{T}$ from equation (2.1.14) which are quoted directly:

Property 1 If a>0, c>0 and f≤0 in (2.1.9), then the matrix representation of equation (2.1.11) is *diagonally dominant* for all Δx and Δy with $(\tilde{T}_{i,i})<0$ and $(\tilde{T}_{i,j}) \geq 0$, $i \neq j$. This is a direct consequence of equation (2.1.14) which implies that with appropriate boundary conditions, and a>0, c>0 and f≤0, the finite representation of equation (2.1.11) is unconditionally stable.

Property 2 If the system (2.1.9) is hyperbolic or elliptic, then equation (2.1.11) preserves this hyperbolic or elliptic nature of the system.

Property 1 shows the unconditional convergence of the numerical representation of UDR while property 2 shows that the numerical solution contains the hyperbolic or elliptic characteristic information of the given system. Taylor series-based schemes cannot directly preserve property 1 without adding the numerical viscosity artificially. Nor can they preserve property 2 without employing switching functions.

The most distinguishing feature of the Roscoe discretization process is that it tries to keep as many of the physical properties of the original system as possible. It starts from system simulation by local linearization of the original nonlinear system which results in a set of solvable linear systems (at least in the one dimension case) on each subdomain. Consequently, an analytical solution to this linear system can be found for every interval and then a discretized algebraic equation is constructed on these analytical solutions. In this way, the numerical solution reflects the local information in the real system. Thus, it

is possible to learn more about the system by analyzing the numerical solution since the characteristic properties are contained in it.

Following similar ideas, other researchers in CFD have continued Roscoe's work. El-Mistikawy and Werle [3] (1977) found in their numerical study for inviscid/viscous fluid flows that the exponential operator in the discrete representation of the PDEs removes unphysical oscillatory behavior which is inherent in many finite difference methods. Instead of using the nodal point values for the nonlinear coefficient functions f_1, f_2, they used the middle point values on $[x_{i-1},x_i]$ and $[x_i,x_{i+1}]$ which resulted in a Exponential Box scheme. This Exponential Box scheme was found to be second order accurate in both viscous and inviscid flows.

Numerically, Roscoe discretization gives an unconditionally stable algebraic system. The method so far works for arbitrary cell Reynolds number with good results which makes it possible to explore some difficult fluid dynamic problems. From the previous discussion, it is known that for a Roscoe type scheme, the numerical error is due to approximating the coefficients f_1, f_2 and f_3 by local constants (at least for 1-D problems). This implies that the accuracy might be improved if those coefficients are approximated to a higher order.

2.2 Generalization of the Roscoe scheme

It is common practice to treat the governing equations individually. In each equation, the terms are distinguished by their physical interpretation as diffusion, convection or pressure gradient term and are approximated individually. In constructing the numerical models, the discretized expressions are created separately for each equation in the system by approximating derivatives term by term. The problem with this discretization process is that the original system interactions among the terms and equations in the original system are ignored. Thus when the finite difference system is reassembled, the interactions between the terms have been replaced by the superposition of the functions represented by the term by term discretization The original interactions can be retained if the problem is studied as a system instead of as individual terms or equations.

In the present study, a preliminary systematic implementation of a generalization of the Roscoe scheme is developed and applied to solving the one-dimensional *steady state* Burgers equation (Chapter 3) and two-dimensional *unsteady* Lamb Vortex problem (Chapter 4). Both of these test problems have exact solutions that may be used to evaluate the accuracy of the numerical solutions.

2.2.1 Problems with the original Roscoe scheme

The original Roscoe scheme presented a discretization of convection-diffusion pairs and thus preserves some interaction between these types of terms. This will be shown below followed by a discussion of what that method leaves out.

Consider the one-dimensional nonlinear Burgers equation which is given by:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \varepsilon\frac{\partial^2 u}{\partial x^2} = 0 \tag{2.2.1}$$

This equation can be written in the following form:

$$a\frac{\partial u}{\partial x} - b\frac{\partial^2 u}{\partial x^2} = r \tag{2.2.2}$$

where $a = u(x,t)$, $b = \varepsilon$ and $r = -\partial u/\partial t$. Applying Taylor expansion on both the 'a' and the 'r' coefficients gives:

$$\begin{aligned} a = u(x,t) &= u_i + \left(\frac{\partial u}{\partial x}\right)_i \Delta x + \left(\frac{\partial u}{\partial t}\right)_i \Delta t + \ldots \\ r = r(x,t) &= -\frac{\partial u}{\partial t} = -\left[\left(\frac{\partial u}{\partial t}\right)_n + \left(\frac{\partial^2 u}{\partial t \partial x}\right)_n \Delta x + \left(\frac{\partial^2 u}{\partial t^2}\right)_n \Delta t + \ldots\right] \end{aligned} \tag{2.2.3}$$

In Roscoe's analysis, only the first terms in equation (2.2.3) are kept, that is, when the constant local values of u and r are used to approximate u and $\partial u/\partial t$, equation (2.2.2) then has the following analytical solution for u:

$$\begin{aligned} u(x) &= c_1 + c_2 x + c_3 e^{\lambda \Delta x} \\ \lambda &= \frac{a}{b} \\ c_2 &= \frac{r}{a} \end{aligned} \tag{2.2.4}$$

where c_1 and c_3 are integration constants. Note that u(x) in the expression of equation (2.2.4) is a combination of a linear polynomial and an exponential term. In addition,

$r = -\partial u/\partial t$ is approximated by a piecewise constant function in both space and time. The importance of this will be discussed later.

On a discrete calculation domain with a uniform grid size h, the constants c_1, c_2 and c_3 can be determined by requiring equation (2.2.4) to pass through the nodal points x_{i-1}, x_i and x_{i+1} on the subdomain $[x_{i-1}, x_{i+1}]$. Then, r can be solved directly in terms of node point values:

$$r = -u_i\left[\frac{e^{\lambda_i \Delta x}u_{i-1} - (1+e^{\lambda_i \Delta x})u_i + u_{i+1}}{(-1+e^{\lambda_i \Delta x})\Delta x}\right] \qquad (2.2.5)$$

$$\lambda_i = \frac{u_i}{\varepsilon}$$

Comparing equation (2.2.5) with equation (2.2.2) shows that the right hand side of equation (2.2.5) represents the spatial differencing of both the convection and diffusion terms.

At the zero velocity point where $u \to 0$, the expression (2.2.5) has a well defined limit given by:

$$r = -\frac{\varepsilon(u_{i-1} - 2u_i + u_{i+1})}{\Delta x^2} \quad \text{with } \lambda_i = \frac{u_i}{\varepsilon} \to 0 \qquad (2.2.6)$$

which is seen to be the second order central difference expression for the diffusion term. This is a correct limit since, as $u \to 0$, the convection term $u(\partial u/\partial x)$ vanishes and Burgers equation simply becomes the heat conduction equation: $\frac{\partial u}{\partial t} - \varepsilon\frac{\partial^2 u}{\partial x^2} = 0$. Applying Roscoe's method gives an analytical solution which is a quadratic polynomial: $u(x) = \frac{r}{2}x^2 + c_1 x + c_2$. Its second derivative in a discrete formulation is given in equation (2.2.6).

For the time derivative, Roscoe and the authors of [3, 10, 11, 12] all used the backward Euler form:

$$\frac{\partial u}{\partial t} = \frac{u^{n+1} - u^n}{\Delta t} = -r \qquad (2.2.7)$$

The error introduced during the above process for this one dimension problem comes from the first order linearization according to equation (2.2.3) by evaluating the velocity u and its time derivative $\partial u/\partial t$ with local constant values (the leading truncated terms in equation (2.2.3) are of the order of Δx and Δt respectively).

A question arises here: if a higher order approximation for u and r is applied, that is, if linear or higher order interpolate functions are adopted for u and r instead of the constant local values, will this method give higher accuracy? This question has not been fully examined in present work. But, in the study of the generalized Roscoe scheme, an effort has been made to start such an investigation. In equation (2.2.3), if u is still approximated with the local constant value, but, r, is approximated by the first two terms in the expansion so that r is locally a linear function of x, then:

$$r = \alpha x + \beta \tag{2.2.8}$$

where α, β are some arbitrary constants. This is a second order linearization for r at least for the spatial derivative x since the truncated terms then have orders of $\Delta x^2, \Delta x^3, \ldots$, etc. with respect to x. With this new approximation for r, the analytical solution to equation (2.1.2) changes to:

$$u(x) = c_1 + c_2 x + c_3 x^2 + c_4 e^{\lambda x} \tag{2.2.9}$$

Comparing to solution (2.1.4), u(x) in above expression becomes a combination of a quadratic polynomial and an exponential term. The local information of the original system is still included via the parameter $\lambda = U/\varepsilon$.

For systems of equations with only convection-diffusion pairs, the Roscoe procedure gives a complete differencing methodology with (as will be shown) superior accuracy and stability over Taylor series-based methods of the same order. However, the procedure gives no guidance for other derivative types. To see this consider the two dimensional incompressible flow equations in Cartesian coordinates:

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0$$

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}=\frac{1}{Re}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)-\frac{\partial p}{\partial x} \qquad (2.2.10)$$

$$\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}=\frac{1}{Re}\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right)-\frac{\partial p}{\partial y}$$

In the original Roscoe scheme, convection-diffusion pairs in the x-momentum equation are integrated to give interpolating functions for u. Similarly, integration of the convection-diffusion pairs in the y-momentum equation yields interpolating functions for v. By rearranging the momentum equations, system (2.2.10) can be written as:

$$u\frac{\partial u}{\partial x}-\frac{1}{Re}\frac{\partial^2 u}{\partial x^2}=r_1$$
$$v\frac{\partial u}{\partial y}-\frac{1}{Re}\frac{\partial^2 u}{\partial y^2}=R_1 \qquad (2.2.11a)$$

$$u\frac{\partial v}{\partial x}-\frac{1}{Re}\frac{\partial^2 v}{\partial x^2}=r_2$$
$$v\frac{\partial v}{\partial y}-\frac{1}{Re}\frac{\partial^2 v}{\partial y^2}=R_2 \qquad (2.2.11b)$$

with:

$$r_1=\frac{1}{Re}\frac{\partial^2 u}{\partial y^2}-\frac{\partial p}{\partial x}-v\frac{\partial u}{\partial y}-\frac{\partial u}{\partial t}$$
$$R_1=\frac{1}{Re}\frac{\partial^2 u}{\partial x^2}-u\frac{\partial u}{\partial x}-\frac{\partial p}{\partial x}-\frac{\partial u}{\partial t} \qquad (2.2.12a)$$

$$r_2=\frac{1}{Re}\frac{\partial^2 v}{\partial y^2}-\frac{\partial p}{\partial y}-v\frac{\partial v}{\partial y}-\frac{\partial v}{\partial t}$$
$$R_2=\frac{1}{Re}\frac{\partial^2 v}{\partial x^2}-u\frac{\partial v}{\partial x}-\frac{\partial p}{\partial y}-\frac{\partial v}{\partial t} \qquad (2.2.12b)$$

where r_1, r_2, R_1, and R_2 are taken as constants in the Roscoe scheme. Equations (2.2.11) and (2.2.12) are ordinary differential equations for u and v respectively. They fit into the form of equation (2.1.1) for which the Roscoe discretization method can be applied. Discretized representations of the x and y momentum equation for u and v are thus obtained individually; they are solved separately for u and v.

Thus the convection-diffusion pairs were integrated to give interpolating functions for u and v. The process, though, ignored the pressure gradient and the continuity equation. In the original Roscoe work, the continuity equation and pressure gradient terms were treated with Taylor expansion based finite difference approximations.

There are three problems that emerge here:

(i) The inconsistency in discretization of the pressure terms and of the continuity equation.

(ii) The interaction between the original continuity equation, x-momentum equation and y-momentum equation is changed from the governing equations (just like any other finite difference procedure). This is due to deriving interpolants from only one equation. Thus the interpolants for u, for example, only come from the x-momentum equation and have no contribution from the y-momentum or continuity equations.

(iii) Approximating the forcing functions r_1, r_2, R_1, and R_2 as constants in equation (2.2.11) introduces errors because these quantities are actually functions of the both time and spatial variables.

In the Roach and Calhoon-Roach [11,12] version of the Roscoe scheme, the first problem was addressed by keeping the pressure gradient terms on the left hand side (inside the integration) rather than as constants in equation (2.2.11). The idea was to change from integration of convection-diffusion pairs in one direction to "all terms" in one direction. Thus the pressure gradient terms were added to the appropriate convection-diffusion pair in the same direction. With this approach, equation (2.2.11) becomes:

$$u\frac{\partial u}{\partial x} - \frac{1}{Re}\frac{\partial^2 u}{\partial x^2} + \frac{\partial p}{\partial x} = r_1$$
$$v\frac{\partial u}{\partial y} - \frac{1}{Re}\frac{\partial^2 u}{\partial y^2} = R_1 \qquad (2.2.13a)$$

$$u\frac{\partial v}{\partial x} - \frac{1}{Re}\frac{\partial^2 v}{\partial x^2} = r_2$$
$$v\frac{\partial v}{\partial y} - \frac{1}{Re}\frac{\partial^2 v}{\partial y^2} + \frac{\partial p}{\partial y} = R_2 \qquad (2.2.13b)$$

with:

$$r_1 = \frac{1}{Re}\frac{\partial^2 u}{\partial y^2} - v\frac{\partial u}{\partial y} - \frac{\partial u}{\partial t}$$
$$R_1 = \frac{1}{Re}\frac{\partial^2 u}{\partial x^2} - u\frac{\partial u}{\partial x} - \frac{\partial u}{\partial t} \qquad (2.2.14a)$$

$$r_2 = \frac{1}{Re}\frac{\partial^2 v}{\partial y^2} - v\frac{\partial v}{\partial y} - \frac{\partial v}{\partial t}$$
$$R_2 = \frac{1}{Re}\frac{\partial^2 v}{\partial x^2} - u\frac{\partial v}{\partial x} - \frac{\partial v}{\partial t} \qquad (2.2.14b)$$

In the integration, a linear interpolation for pressure was used. This is an improvement over the original Roscoe method, although the linear relation for pressure was something assumed rather than being a property of the system of equations. However, the interpolating functions for the velocity components were still derived by integrating all derivatives in one direction in a single momentum equation. Thus the x-momentum equation gave the interpolating functions for the u-component of velocity, the y-momentum equation gave the interpolating functions for the v-component. Hence, like the original Roscoe procedure, the interpolating functions for the velocities arose only from the momentum equations which contains no information from the other equations in the system when the quantities r_1, r_2, R_1, and R_2 are taken as constants.

2.2.2 A Systematic Approach to the Roscoe Scheme

In the generalization, the problem is solved as a system. The interpolating functions for all the dependent variables are found in a coupled fashion by integrating all derivatives in a given direction for the entire system of linearized equations. Since derivatives in the other directions are placed in a right hand side vector, the mathematical technique is that of finding the solution to a *system* of non-homogenous ordinary differential equations. Thus the interpolating functions for the velocity components will have a contribution from all equations in the system, not just a single momentum equation. In doing so, system (2.2.10) becomes two ordinary differential systems:

$$\frac{\partial u}{\partial x} = r_1$$
$$u\frac{\partial u}{\partial x} - \frac{1}{Re}\frac{\partial^2 u}{\partial x^2} + \frac{\partial p}{\partial x} = r_2 \qquad (2.2.15)$$
$$u\frac{\partial v}{\partial x} - \frac{1}{Re}\frac{\partial^2 v}{\partial x^2} = r_3$$

$$\frac{\partial v}{\partial y} = R_1$$

$$v\frac{\partial u}{\partial y} + \frac{1}{Re}\frac{\partial^2 u}{\partial y^2} = R_2 \qquad (2.2.16)$$

$$v\frac{\partial v}{\partial y} + \frac{1}{Re}\frac{\partial^2 v}{\partial y^2} + \frac{\partial p}{\partial y} = R_3$$

The vectors $\vec{r}^T = (r_1, r_2, r_3)$ and $\vec{R}^T = (R_1, R_2, R_3)$, which for convenience are referred herein as Roscoe vectors, will need to be defined. If the original Roscoe procedure is followed, they would all be constant. This will be discussed below.

Note that the procedure has taken the two dimensional system (2.2.10) and separated it into two one-dimensional ordinary differential equation systems (2.2.15) and (2.2.16) which can be solved by an obvious extension of the Roscoe scheme. Since each set of equations is solved as a system, the dependent variables u, v and p will have the correct dynamic influences from all the equations contained in the system. The influences from the y direction are still present since they are inherent to the vector $\vec{r}^T = (r_1, r_2, r_3)$. After linearization in the same fashion as in section 2.1, system (2.2.15) becomes, in matrix form:

$$\begin{pmatrix} 1 & 0 & 0 \\ U & 0 & 1 \\ 0 & U & 0 \end{pmatrix} \frac{\partial}{\partial x} \begin{pmatrix} u \\ v \\ p \end{pmatrix} - \begin{pmatrix} 0 & 0 & 0 \\ \varepsilon & 0 & 0 \\ 0 & \varepsilon & 0 \end{pmatrix} \frac{\partial^2}{\partial x^2} \begin{pmatrix} u \\ v \\ p \end{pmatrix} = \vec{r} \qquad (2.2.17)$$

where U is some constant velocity and $\varepsilon = 1/Re$. Integrating this second order system once gives the first order system:

$$\begin{pmatrix} 1 & 0 & 0 \\ U & 0 & 1 \\ 0 & U & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ p \end{pmatrix} - \begin{pmatrix} 0 & 0 & 0 \\ \varepsilon & 0 & 0 \\ 0 & \varepsilon & 0 \end{pmatrix} \frac{\partial}{\partial x} \begin{pmatrix} u \\ v \\ p \end{pmatrix} = \int \vec{r} dx + \vec{C} \equiv \vec{s} + \vec{C} \qquad (2.2.18)$$

for some constant vector $\vec{C}$. It is inherent in this procedure that the interpolating functions for each of the variables are linked together. That is, the resulting functions from the integration are influenced by the entire set of equations. This is in contrast to all the previous versions of the Roscoe method where only a single momentum equation

gives rise to an interpolating function. This is certainly in stark contrast to Taylor series methods where there is no attempt to build in any satisfaction of a even single equation.

The simplified system of incompressible flow equations (2.2.18) is singular since it is not possible to rewrite it in the form:

$$\frac{\partial}{\partial x}\begin{pmatrix} u \\ v \\ p \end{pmatrix} = \mathbf{A}\begin{pmatrix} u \\ v \\ p \end{pmatrix} + \bar{\mathbf{B}} \tag{2.2.19}$$

where $\mathbf{A}$ is a 3×3 matrix. That is, the coefficient matrix in equation (2.2.18) that multiples the derivative term is not invertible. However, the system described by equation (2.2.18) is tractable in the sense of Cambell and Meyer [13] and may be integrated by means of the Drazin generalized inverse, Moulden (unpublished UTSI report) applies this technique to give:

$$\begin{pmatrix} u(x) \\ v(x) \\ p(x) \end{pmatrix} = \begin{pmatrix} u(x_0) + \int_{x_0}^{x} r_1(t)dt \\ u(x_0)e^{-\alpha(x-x_0)} - e^{-\alpha x}\left[\int_{x_0}^{x} s_3(t)e^{\alpha t}/\varepsilon dt + s_{03}\left(1-e^{\alpha(x-x_0)}\right)/U\right] \\ p(x_0) + \int_{x_0}^{x}[r_2(t) - Ur_1(t)]dt + \varepsilon[r_1(x) - r_1(x_0)] \end{pmatrix} \tag{2.2.20}$$

Following equation (2.2.18), the above procedure is repeated in the y-direction, and a similar solution is obtained as:

$$\begin{pmatrix} u(y) \\ v(y) \\ p(y) \end{pmatrix} = \begin{pmatrix} u(y_0)e^{-\beta(y-y_0)} - e^{-\beta y}\left[\int_{y_0}^{y} S_2(t)e^{\beta t}/\varepsilon dt + S_{02}\left(1-e^{\beta(y-y_0)}\right)/V\right] \\ v(y_0) + \int_{y_0}^{y} R_1(t)dt \\ p(y_0) + \int_{y_0}^{y}[R_3(t) - VR_1(t)]dt + \varepsilon[R_1(y) - R_1(y_0)] \end{pmatrix} \tag{2.2.21}$$

where s_{02} and s_{03} are constants of integration evaluated at x_0 or y_0 respectively.

It is left, now, to fix the spatial dependence of the vectors $\vec{r}$ and $\vec{R}$. When they are selected as:

$$\vec{r} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \begin{pmatrix} \alpha_1 + \beta_1 e^{Ux/\varepsilon} \\ constant \\ constant \end{pmatrix} \tag{2.2.22}$$

$$\vec{R} = \begin{pmatrix} R_1 \\ R_2 \\ R_3 \end{pmatrix} = \begin{pmatrix} \gamma_1 + \delta_1 e^{Vy/\varepsilon} \\ constant \\ constant \end{pmatrix} \tag{2.2.23}$$

with $\vec{s} = \int \vec{r}(x)dx$ and $\vec{S} = \int \vec{R}(x)dx$, it is found from solution (2.2.20) that each of the interpolation functions has the same generic form given as:

$$\phi = a + bx + ce^{\lambda x} \tag{2.2.24}$$

with a similar form in the y-direction. This is the same form that Roscoe obtained for the velocity components by simply appealing to the momentum equations. What's new is that here it is found that the pressure also has the same form. In Roscoe's original work and in the work of Roach and Calhoon, it was not known what form the pressure interpolation should take. In addition, equations (2.2.20) and (2.2.21) link the coefficients of individual interpolation functions. It should be noted, however, there is no rational reason for the coefficients, α_1 and β_1, in equations (2.2.22) and γ_1, δ_1 in equation (2.2.23).

Similarly, when it is assumed that:

$$\vec{r} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \begin{pmatrix} \alpha_1 + \beta_1 x + \gamma_1 e^{Ux/\varepsilon} \\ \alpha_2 + \beta_2 x \\ constant \end{pmatrix} \tag{2.2.25}$$

and

$$\vec{R} = \begin{pmatrix} R_1 \\ R_2 \\ R_3 \end{pmatrix} = \begin{pmatrix} \eta_1 + \delta_1 y + \sigma_1 e^{Vy/\varepsilon} \\ \text{constant} \\ \eta_3 + \delta_3 y \end{pmatrix} \tag{2.2.26}$$

the generic form for the interpolation function is:

$$\phi = a + bx + cx^2 + de^{\lambda x} \tag{2.2.27}$$

with a similar y-direction equation. These are the quadratic exponential interpolation functions obtained before.

The form of the particular solution depends upon the specification of the Roscoe vectors $\vec{r}$ and $\vec{R}$. So one cannot expect a general form with this method, instead, there are different solutions from system (2.2.18) for different definitions of $\vec{r}$ and $\vec{R}$. Since the original system is second order, each variables u, v, p can only take two constants that are obtained to satisfy the natural boundary conditions of the problem. The extra constants come from the Roscoe vectors and allow additional constraints to be imposed upon the interpolation functions. As will be shown in section 5.1, these extra constraints do not change the spatial order of the method. In the generalized Roscoe scheme, the Roscoe vectors $\vec{r}$ and $\vec{R}$ are approximated with functions of the spatial variables instead of constants and this decreases the simulation error. Since the study of the Generalized Roscoe scheme is an ongoing work, the solution to equation (2.2.18) has not been obtained for all possible variations of $\vec{r}$ and $\vec{R}$. Nor has a rational technique for finding these vectors been developed. In the present work, two versions of the generalized Roscoe scheme are developed. The one which is based on solution (2.2.24) is referred to as the *Linear Exponential Spline* method while the other one, the *Quadratic Exponential Spline* method is based on solution (2.2.27).

2.2.3 The Spline Interpolation

In the present work, two schemes derived from the generalized Roscoe method are developed by applying spline interpolations. From a function approximation point of view, comparing with the piecewise polynomial approximation, the polynomial spline interpolation has some better properties, and thus gives the best approximation as long as the interpolation function is an odd degree polynomial. Many kinds of spline interpolates have been used to obtain approximate solutions to ordinary differential equations[14].

The linear exponential and quadratic exponential splines mentioned above fall into a broader category of so called generalized splines. The existence, uniqueness and convergence of the solutions to differential equations using these generalized splines have been proved [15]. Like the odd degree polynomial splines, the generalized splines also have the minimum property and then consequently give the best approximation in the appropriate norm. In another words, once the approximating functions have been selected, the best results can be expected from spline interpolation.

From sections 2.1 and 2.2, it is now known that for the generalized Roscoe scheme, the local analytical solutions have the form of either equation (2.2.24) or equation (2.2.9). To take advantage of this generality and to approach the best approximation, a spline interpolation with these linear or quadratic exponential functions as the interpolation functions is applied in the solving procedure. When the interpolation is appropriately set up, the resulting matrices for solving the spline coefficients are diagonally dominant which guarantee a convergent numerical procedure for determining these spline coefficients. Furthermore, the derivatives needed in the time marching process can be evaluated from direct differentiation of the spline function at each node point. This direct application of spline function shows better accuracy than do the other second-order accurate schemes (see Figure 5.1.2, for example).

Corresponding to different simplifications of the original differential equations, two kinds of spline have been used in evaluating the interpolation functions for the unknown dependent variables. In the following definitions, the individual interpolation functions will be referred to by the symbol $s_i(x)$ while the total spline generated from a collection of $s_i(x)$ will be denoted as S:

Definition 2.2.1 Given a function f(x) defined on [a, b] and a set of numbers, called nodes, $a = x_1 < x_2 < \cdots < x_{i-1} < x_i < x_{i+1} \cdots < x_{n-1} < x_n = b$ with a uniform grid size $h = x_{i+1} - x_i = x_i - x_{i-1}$ for simplicity, a *linear exponential interpolant,* S, for f is a function that satisfies the following conditions:

(a) S is a linear exponential function with the form of (2.2.14), denoted as s_i on the subintervals $[x_i - h/2, x_i + h/2]$ for each $i = 3,...,n-2$ and $[x_2 - h, x_2 + h/2]$, $[x_{n-1} - h/2, x_{n-1} + h]$ for the two end points, see Figure 2.2.1;

$$s_i(x) = a_i + b_i x + c_i e^{\lambda_i x} \tag{2.2.28}$$

where $\lambda_i \neq 0$ are some constants related to the differential equation.

(b) $s(x_i) = u_i$ for each i = 1,,2, ... n;

(c) $s_{i+1}(x_{i+1} - h/2) = s_i(x_i + h/2)$ for each i = 2, 3, ..., n-2;

(d) $s'_{i+1}(x_{i+1} - h/2) = s'_i(x_i + h/2)$ for each i = 2, 3, ..., n-2.

In above definition, the condition (b) requires the spline function to pass through all the node points. The conditions (c) and (d) require the adjacent spline interpolants to match their values and first derivatives at the matching points.

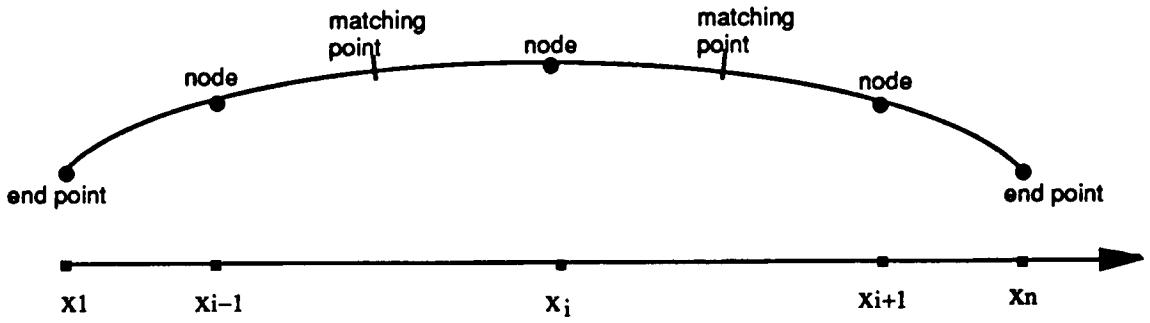


Figure 2.2.1 Definition of the Linear Exponential Spline

Definition 2.2.2 Given a function f(x) defined on [a, b] and a set of numbers, called nodes, $a = x_1 < x_2 < \cdots < x_{i-1} < x_i < x_{i+1} \cdots < x_{n-1} < x_n = b$ with a uniform grid size $h = x_{i+1} - x_i = x_i - x_{i-1}$ for simplicity, a *quadratic exponential interpolant,* S, for f(x) is a function that satisfies the following conditions:

(a) S is a quadratic exponential function, denoted s_i, on the intervals $[x_i, x_{i+1}]$, for each i=2,3,,,...,n-1, see Figure 2.2.2:

$$s_i(x) = a_i + b_i x + c_i x^2 + d_i e^{\lambda_i x} \qquad (2.2.29)$$

where $\lambda_i \neq 0$ are some constants that come from the differential equation.

(b) $s(x_i) = u_i$, i = 1,,2, ... n;

(c) $s_{i+1}(x_i) = s_i(x_i)$, i = 2, 3, ..., n-1;

(d) $s'_{i+1}(x_i) = s'_i(x_i)$, i = 2, 3, ..., n-1;

(e) $s''_{i+1}(x_i) = s''_i(x_i)$, i = 2, 3, ..., n-1;

(f) one of the following set of boundary conditions is introduced:

(i) $s''(x_1)$ and $s''(x_n)$ are specified, or,

(ii) $s'(x_1)$ and $s'(x_n)$ are specified.

In definition 2.2.2, the condition (b) requires the spline function to pass through all the node points. The conditions (c), (d) and (e) require the adjacent spline interpolants to match their values as well as their first and second derivatives at the matching points.

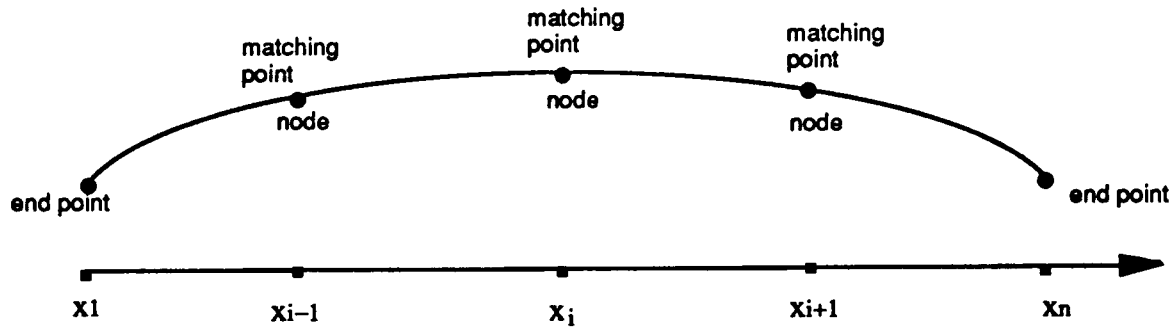


Figure 2.2.2 Definition of the Quadratic Exponential Spline

2.3 Phase space study for the Roscoe linearization

The theory of differential equations is naturally stated in terms of a system of first order equations. The above generalization of the Roscoe method is also stated this way. The vector $(u,v,p)^T$ arising in equation (2.2.18) is defined to be a point in *phase space* for the system of equations. The generalized Roscoe method thus has an interpretation in phase space and it makes sense to interpret the numerical methods in this space also. The following is a brief introduction to this interpretation process.

In section 2.2 it was shown that a spatial discretization error in the generalized Roscoe methods is introduced during the linearization of the nonlinear coefficients of the differential equation. It is instructive to show how this linearization influences the system behavior. Hence, the steady state phase space for Burgers equation, $\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \varepsilon\frac{\partial^2 u}{\partial x^2}$, has been studied. It is found that, for the Burgers equation, the phase space of the linearized system preserves the stability property of the nonlinear system and the approximation error generated in the linearization can be estimated from the phase space slope and curvature expressions. Burgers equation will be studied in detail in Chapter 3.

2.3.1 The phase space for the nonlinear Burgers equation

The steady state Burgers equation is given by:

$$u\frac{\partial u}{\partial x}-\varepsilon\frac{\partial^2 u}{\partial x^2}=0 \qquad (2.3.1)$$
$$u(-\infty)=1;\ \ u(\infty)=-1$$

By changing variables as

$$v_1=u$$
$$v_2=v_1'=\frac{\partial u}{\partial x} \qquad (2.3.2)$$
$$v_2'=\frac{\partial^2 u}{\partial x^2}$$

a system of equations:

$$v_2=v_1'$$
$$v_1v_2=\varepsilon v_2' \qquad (2.3.3)$$

is generated.

Dividing the second equation in system (2.3.3) by the first gives the phase space equation:

$$\frac{dv_2}{dv_1}=\frac{1}{\varepsilon}\frac{v_1v_2}{v_2}=\frac{1}{\varepsilon}v_1 \qquad (2.3.4)$$

with $v_2=0$ as a singular line. Integrating equation (2.3.4) gives:

$$v_2=\frac{v_1^2}{2\varepsilon}+\frac{c}{\varepsilon} \qquad (2.3.5)$$

where c is the integration constant which can be determined from the boundary conditions known from physical space. As in the analytical solution, take $v_1=u=\pm1$, $v_2=v_1'=0$, when $c=-1/2$. Thus, equation (2.3.5) becomes:

$$v_2=\frac{v_1^2-1}{2\varepsilon} \qquad (2.3.6)$$

which shows that when $|v_1|\le 1$, there is $v_2\le 0$. This means that, on the phase space, only the segment of phase space with $v_2\le 0$ is of real interest. However, the line $v_2=0$ is a singular line as noted above. The singular points are unstable for $v_1>0$ and stable

for $v_1 < 0$ as can readily be checked. With equation (2.3.6), the phase space for the exact Burgers equation can be drawn as shown in Figure 2.3.1:

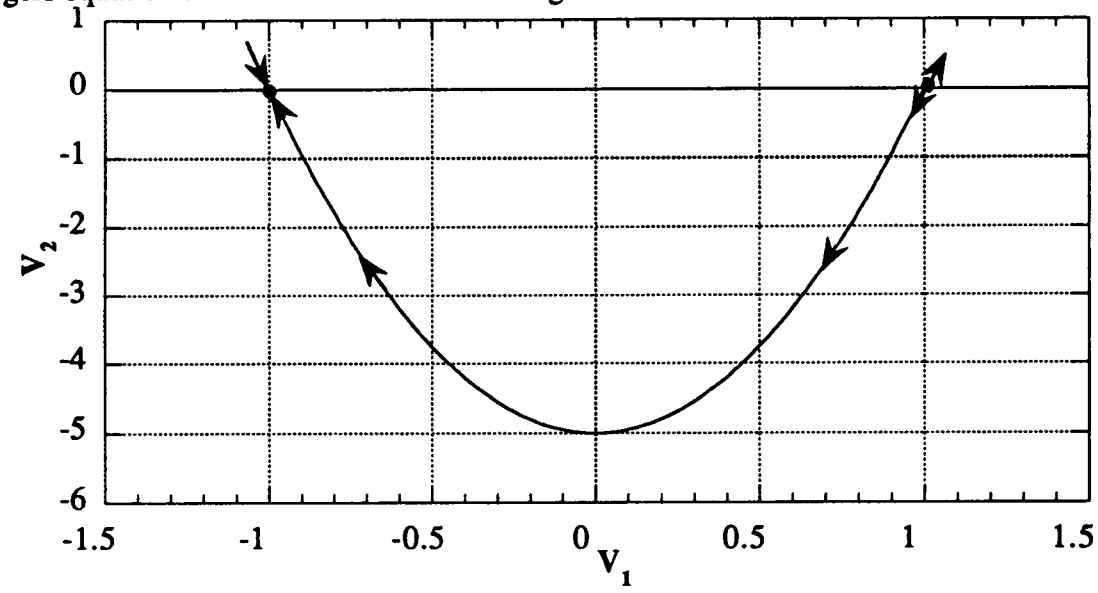


Figure 2.3.1 Phase Space for the Nonlinear Burgers Equation

2.3.2 Phase space for the locally linearized Burgers equation

Linearization of Burgers equation (3.1), gives:

$$U\frac{\partial u}{\partial x} - \varepsilon\frac{\partial^2 u}{\partial x^2} = r \tag{2.3.7}$$

where U is some constant velocity. Using the same new variables as those defined in equation (2.3.2), a system is obtained:

$$\begin{aligned} v_2 &= v_1' \\ Uv_2 &= \varepsilon v_2' + r \end{aligned} \tag{2.3.8}$$

In matrix form, this becomes:

$$\begin{pmatrix} v_1 \\ v_2 \end{pmatrix}' = \begin{pmatrix} 0 & 1 \\ 0 & U/\varepsilon \end{pmatrix}\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} + \begin{pmatrix} 0 \\ -r/\varepsilon \end{pmatrix} \tag{2.3.9}$$

or, in vector form:

$$\vec{v}' = A\vec{v} + \vec{b} \tag{2.3.10}$$

with:

$$\vec{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \quad A = \begin{pmatrix} 0 & 1 \\ 0 & U/\varepsilon \end{pmatrix} \text{ and } \vec{b} = \begin{pmatrix} 0 \\ -r/\varepsilon \end{pmatrix} \tag{2.3.11}$$

It may be noted here that the coefficient matrix A has eigenvalues zero and U/ε. So for large U/ε, the equation is stiff in the usual numerical analysis sense. The equation is degenerate as $U/\varepsilon \to 0$ with only one true eigenvector, $(1,0)^T$, corresponding to the repeated eigenvalue 0.

Applying the technique from differential equation theory, the homogenous solution to equation (2.3.10) is obtained as:

$$v_2 = \frac{U}{\varepsilon}(v_1 - \gamma) \tag{2.3.12}$$

where $\gamma = v_1(0) - \frac{\varepsilon}{U} v_2(0)$ is a constant. These are straight lines with slope given by U/ε as shown in Figure 2.3.2

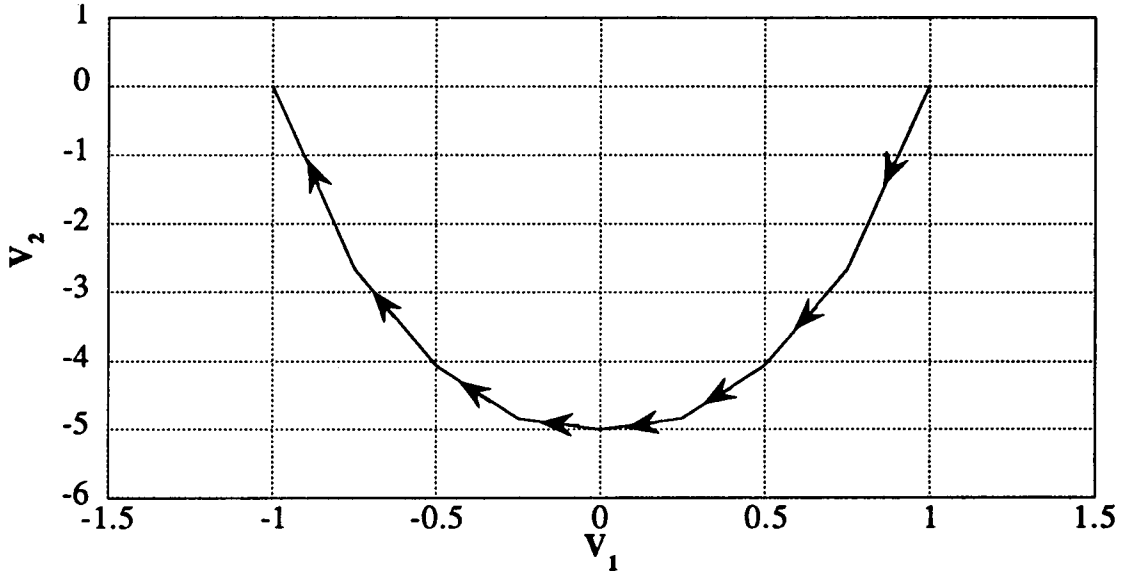


Figure 2.3.2 Phase Space for the Locally Linearized Burgers Equation

The second derivative term represents the slope of the phase space curve in Fig. 2.3.1 and Fig. 2.3.2 since:

$$v'' = \frac{dv_2}{dx} = \frac{dv_2}{dv_1}\frac{dv_1}{dx} = v_2 \frac{dv_2}{dv_1} \tag{2.3.13}$$

From equation (2.3.6), the slope of phase space for the nonlinear system is found to be a linear function of v_1,

$$\frac{dv_2}{dv_1} = \frac{v_1}{\varepsilon} \tag{2.3.14}$$

From equation (2.3.12), the slope of the phase space for the linearized system is a constant:

$$\frac{dv_2}{dv_1} = \frac{U}{\varepsilon} = \lambda \tag{2.3.15}$$

and is collinear with the exact phase space slope at the selected node point. However, the constant λ is changing with the node point, i.e. $\lambda_i = U_i/\varepsilon$, when it is applied to the discretized expression. U_i is the approximation of $v_{1_i} = v_1(x_i)$ according to equation (2.2.3). Obviously, when the grid size tends to zero, U_i tends to v_1 everywhere and the phase space of the linearized system tends to that of the nonlinear system. From control theory, it is known that the stability status of a system is determined by the homogenous part of the system. Therefore, from Figure 2.3.1, it is seen that if the system (2.3.1) starts from a point where $v_2 < 0$, it will go to a stable fixed point on the line $v_2 = 0$ along the parabola (2.3.6). The linearized system in equation (2.3.7) also goes to the stable fixed point if it starts from the same place, however, it follows a straight line on each subdomain with different slopes $\lambda_i = U_i/\varepsilon$. When the grid size goes to zero, these discrete straight lines recover the parabola of the nonlinear case, then, the phase space for the linearized system tends to that of the nonlinear system. Thus, it may be concluded that, after the linearization in step (2.3.9), the discretized linear system still has the stability of the nonlinear system assuming the grid size to be small.

A study by Moulden (unpublished UTSI report which gives an outline of the above development) shows that the phase space for the approximated system generated by the central difference method does not reflect that of the analytical results at all. From the results of a numerical study and the analysis in the following chapters, it will be seen that the solutions from all the Roscoe type schemes have unconditional stability, while other second order schemes give oscillatory solutions above certain cell Reynolds numbers. Further, if the cell Reynolds number is increased much beyond these certain values, the solving process diverges. Generally, it is very important for a discretization process to conserve the stability property of the original differential system. In the Roscoe discretization, this stability property transfers from the differential equation through the exponential terms with the characteristic roots λ_i, at least in a qualitative way.

The system stability directly relates to the diffusive property within the system. In computational fluid dynamics, diffusion sometimes is defined as a combination of dissipation and dispersion [1]. Since the second derivatives associate with dissipation, then, the accuracy of the slope in the phase space, λ_i, directly influences the dissipation property in the numerical results. The dissipative error is reflected in the second derivative in equation (2.3.13). If equations (2.3.6) and (2.3.12) are substituted into equation (2.3.14), the quantity $v''_{nonlinear}$ is obtained for the nonlinear system. Similarly, when equation (2.3.12) and (2.3.15) are applied, equation (2.3.14) gives v''_{linear}, the second derivative in phase space for the linearized system. The error between the linear and nonlinear second derivatives, $E_{v''} = v''_{nonlinear} - v''_{linear}$, is defined as the local dissipation error generated in the linearization. That is:

$$E_{v''} = \frac{v_1^2}{2\varepsilon^2}(v_1 - U) \tag{2.3.16}$$

This results compares the exact and computed phase spaces on lines of constants v_1.

For the third derivatives, there is:

$$v''' = \frac{d}{dx}\left(\frac{dv_2}{dx}\right) = \frac{d}{dx}\left(v_2\frac{dv_2}{dv_1}\right) = v_1\frac{d}{dv_1}\left(v_2\frac{dv_2}{dv_1}\right) \tag{2.3.17}$$

or:

$$v''' = v_1\left(\left(\frac{dv_2}{dv_1}\right)^2 + v_2\frac{d^2v_2}{dv_1^2}\right) \tag{2.3.18}$$

From equation (2.3.14), the curvature of the phase space, d^2v_2/dv_1^2, for the nonlinear Burgers equation can be obtained as:

$$\frac{d^2v_2}{dv_1^2} = \frac{1}{\varepsilon} \tag{2.3.19}$$

The same variable can be derived for the linearized system from equation (2.3.15):

$$\frac{d^2v_2}{dv_1^2} = 0 \tag{2.3.20}$$

The third derivatives usually relate to the dispersion in the system. It is seen, from equation (2.3.18), that v''' depends upon the curvature as well as the slope of the phase space trajectory. However, the linearized system has zero curvature while the nonlinear equation gives a constant $1/\varepsilon$ for the curvature. These are a direct solutions from equation (2.3.6) and equation (2.3.16). The local dispersion error defined by $E_{v'''} = v'''_{nonlinear} - v'''_{linear}$ can be estimated by calculating $v'''_{nonlinear}$ for the nonlinear system by substituting equation (2.3.6), (2,3.14) and (2.3.19) into equation (2.3.18); v'''_{linear} from equation (2.3.12), (2.3.15) and (2.3.20). Thus, the local dispersion error resulting from the Roscoe linearization can be derived as:

$$E_{v'''} = \frac{v_1}{2\varepsilon^2}\left(3v_1^2 - 2U^2 - 1\right) \tag{2.3.21}$$

If the diffusion error is the combination of the dissipation and the dispersion errors, then, there is:

$$E_{diffusion} = E_{v''} + E_{v'''} = \frac{v_1}{2\varepsilon^2}\left(4v_1^2 - v_1U - 2U^2 - 1\right) \tag{2.3.22}$$

The local diffusion error expression (2.3.22) shows explicitly that the diffusion error introduced in the local linearization is a function of the linearized coefficient U. From equation (2.2.3), it is known that the accuracy of this coefficient U depends on the time step size as well as the spatial grid size. Hence, it can be concluded that the numerical diffusion error depends upon both the time step size and spatial grid size. This conclusion may suggest that better accuracy can be obtained from a higher order approximation in the local linearization of the nonlinear differential equation.

CHAPTER 3

THE STEADY STATE NONLINEAR BURGERS EQUATION

In this chapter, two exponential spline schemes are developed to solve the steady nonlinear Burgers equation.

The nonlinear Burgers equation is given by:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \varepsilon\frac{\partial^2 u}{\partial x^2} = 0 \qquad (3.0.1)$$
$$u(-\infty) = 1; \quad u(\infty) = -1$$

This equation posses a unique smooth solution for all $\varepsilon > 0$ and $t > 0$ (see Kreiss and Lorenz [17]) which satisfies a maximum principle of the form $|u(x,t)|_\infty \le |u(x,0)|_\infty$. Hence, the maximum velocity at time t can never exceed the maximum initial value. A numerical solution which generates a steady state final solution must also satisfy this maximum principle; which thus places restrictions on acceptable computed solutions. The discretization may be consistent and convergent but need not satisfy such a maximum principle: this depends upon the nature of the iteration process expressed symbolically as $\bar{x}_{n+1} = A\bar{x}_n$.

The steady state analytical solution to (3.0.1) is given by:

$$u(x) = a\tanh\left(-\frac{ax}{2\varepsilon}\right) \qquad (3.0.2)$$
$$x \to \pm\infty, \; u \to \pm a$$

where 'a' is a coefficient which represents the reference velocity and can be solved for in an iterative fashion. The reference velocity is dependent upon ε.

As discussed in the previous chapter, the local analytic solution to the locally linearized Burgers equation (3.0.1) given by the Roscoe scheme is:

$$u(x) = a + bx + ce^{\lambda x} \qquad (3.0.3)$$

or, by the generalized Roscoe method, when r is linear in x, there is:

$$u(x) = a + bx + cx^2 + de^{\lambda x} \tag{3.0.4}$$

Consider the nth time level in the computation, that is, when the node point values u_i^n are known, the linear exponential spline or the quadratic exponential spline interpolations can be determined subject to definition 2.2.1 or definition 2.2.2.

3.1 The linear exponential spline interpolation

According to the structure given in definition 2.2.1, the linear exponential spline S is defined in the following manner on the computational domain ordered as $x_1 < x_2 < x_3 < ... < x_i < ... < x_{n-2} < x_{n-1} < x_n$ which is a partition of the physical domain [a,b] with a uniform grid size $h = x_i - x_{i-1}$. The defined domains for s_i are given as: $s_2 \in [x_1, x_2 + h/2]$; $s_i \in [x_i - h/2, x_i + h/2]$, i = 3,4, ..., n−2; $s_{n-1} \in [x_{n-1} - h/2, x_n]$ with:

$$\begin{aligned} s_i &= a_i + b_i x + c_i e^{\lambda_i x} \\ \lambda_i &= \frac{u_i}{\varepsilon} \end{aligned} \tag{3.1.1}$$

There are $n-2$ s_i's and each s_i has three unknown coefficients. The total number of unknowns is $3n-6$. The spline is required to pass through all nodal points and the adjacent s_i and s_{i+1} must match their values and first derivatives at the knot which is at the middle point on each subinterval. More specifically, these conditions are:

(a) $s(x_i) = u_i$ for each i = 1,2, ... n, i.e. n conditions;

(b) $s_{i+1}(x_{i+1} - h/2) = s_i(x_i + h/2)$ for each i = 2, 3, ..., n−2, i.e. n−3 conditions;

(c) $s'_{i+1}(x_{i+1} - h/2) = s'_i(x_i + h/2)$ for each i = 2, 3, ..., n−2, i.e. n−3 conditions.

Overall, there are $3n-6$ conditions. Therefore, it is a closed problem with $3n-6$ linear independent equations. To solve for the a_i's, b_i's and c_i's, first, the a_i's and then the b_i's are eliminated by applying conditions (a), (b) and (c) in a sequence. In this process, x is taken as a parameter on the interval $[x_i - h/2, x_i + h/2]$ instead of being a globally independent variable, that is, the nodal point x_i is taken as the origin (where x=0) for its corresponding subinterval $[x_i - h/2, x_i + h/2]$. This elimination process gives the following equations for the c_i. At the left boundary where i=2, the expression for c_i is:

$$\left[-2 + e^{-\lambda_2 h} + \left(1 + \frac{\lambda_2 h}{2}\right) e^{\frac{\lambda_2 h}{2}}\right] c_2 + \left[1 - \left(1 + \frac{\lambda_3 h}{2}\right) e^{\frac{\lambda_3 h}{2}}\right] c_3 = u_1 - 2u_2 + u_3 \tag{3.1.2}$$

for the interior point where i = 3, ..., n − 2

$$\left[1-\left(1-\frac{\lambda_{i-1}h}{2}\right)e^{\frac{\lambda_{i-1}h}{2}}\right]c_{i-1}+\left[-2+\left(1-\frac{\lambda_i h}{2}\right)e^{-\frac{\lambda_i h}{2}}+\left(1+\frac{\lambda_i h}{2}\right)e^{\frac{\lambda_i h}{2}}\right]c_i+\left[1-\left(1+\frac{\lambda_{i+1}h}{2}\right)e^{-\frac{\lambda_{i+1}h}{2}}\right]c_{i+1}=u_{i-1}-2u_i+u_{i+1} \qquad (3.1.3)$$

while at the right boundary where i = n − 1,

$$\left[1-\left(1-\frac{\lambda_{n-2}h}{2}\right)e^{\frac{\lambda_{n-2}h}{2}}\right]c_{n-2}+\left[-2+e^{\lambda_{n-1}h}+\left(1-\frac{\lambda_{n-1}h}{2}\right)e^{-\frac{\lambda_{n-1}h}{2}}\right]c_{n-1}=u_{n-2}-2u_{n-1}+u_n \qquad (3.1.4)$$

The subsequent matrix generated from the above equations for the c_i's is tridiagonal and can be solved efficiently by a tridiagonal matrix solver. After the c_i's have been determined, the coefficients b_i can be solved easily by a backward substitution:

$$\begin{aligned} b_{n-1} &= \frac{1}{h}\left[u_n-u_{n-1}-c_{n-1}\left(e^{\lambda_{n-1}h}-1\right)\right] \\ b_i &= b_{i+1}+c_{i+1}\lambda_{i+1}e^{-0.5\lambda_{i+1}h}-c_i\lambda_i e^{0.5\lambda_i h} \quad i=2,\ldots, n-1 \\ b_2 &= \frac{1}{h}\left[u_2-u_1-c_2\left(1-e^{-\lambda_2 h}\right)\right] \end{aligned} \qquad (3.1.5)$$

and a_i follows as:

$$a_i = u_i - c_i \qquad (3.1.6)$$

so that the spline passes through the node points.

Thus the global spline function S(x) for u(x) has been found. Consequently, the derivatives $\partial u/\partial x$ and $\partial^2 u/\partial x^2$ at the node points, the quantities of interest, can be approximated by the node point values of $\partial S/\partial x$ and $\partial^2 S/\partial x^2$ which are given by:

$$\left(\frac{\partial u}{\partial x}\right)_i = \left(\frac{\partial S}{\partial x}\right)_i = \left(b_i + c_i\lambda_i e^{\lambda_i x}\right)_i \qquad (3.1.7)$$

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_i = \left(\frac{\partial^2 S}{\partial x^2}\right)_i = \left(c_i \lambda_i^2 e^{\lambda_i x}\right)_i \tag{3.1.8}$$

To be consistent, at the nodal point x_i, x=0. Hence, from the above approximations the derivatives at the nodal points are:

$$\left(\frac{\partial u}{\partial x}\right)_i = b_i + c_i \lambda_i \tag{3.1.9}$$

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_i = c_i \lambda_i^2 \tag{3.1.10}$$

Thus, with b_i, c_i as known values, Burgers equation can be solved by a traditional time marching method. The first order forward time finite difference gives:

$$\frac{\partial u}{\partial t} = \frac{u^{n+1} - u^n}{\Delta t} \tag{3.1.11}$$

Then, the (n+1)th time level values of u_i can be solved explicitly from:

$$u_i^{n+1} = u_i^n - \Delta t\left[-u_i^n\left(b_i + c_i \lambda_i\right) + \varepsilon\left(c_i \lambda_i^2\right)\right] \tag{3.1.12}$$

or implicitly from a delta formulation:

$$\left[-\frac{\varepsilon}{\Delta x^2} + \left(\frac{1}{\Delta t} + \frac{2\varepsilon}{\Delta x^2}\right) - \frac{\varepsilon}{\Delta x^2}\right]\Delta u_i = \left[-u_i\left(b_i + c_i \lambda_i\right) + \varepsilon\left(c_i \lambda_i^2\right)\right] \tag{3.1.13}$$

$$\Delta u_i = u_i^{n+1} - u_i^n$$

It might be noted that in this spline interpolation, the subdomain for each piecewise interpolation function $s_i(x)$ is not defined in the conventional way, that is, the spline interpolates do not match at nodal points, instead, they match at the middle point on each subinterval. The advantages of doing this are:

(i) it gives the same number of conditions as that of the unknowns;

(ii) in the exponential spline interpolation, it is important to chose the proper λ value for each interval. It has been found that the middle point λ value is so far the best choice. So, if the spline is defined as in the definition 2.2.2, then $\lambda_i = u_i/\varepsilon$ calculated at the nodal points is automatically the middle point value for each spline interpolant.

3.2 The quadratic exponential spline interpolation

When r in equation (2.2.2) is approximated by a linear function of x instead of a constant, the local analytical solution has the form of equation (2.2.9) which is a quadratic polynomial plus an exponential term. In this case, the spline interpolant s_i has the same structure:

$$s_i(x) = a_i + b_i x + c_i x^2 + d_i e^{\lambda_i x}$$
$$\lambda_i = \frac{u_i(x + h/2)}{\varepsilon} \qquad (3.2.1)$$

which is defined on: $[x_i, x_{i+1}]$ i=1,2,3,...,n−1 according to definition 2.2.2. The number of unknowns is now $4n-4$. The spline is required to pass through the node points and the adjacent s_i and s_{i+1} have to match their first and second derivatives at their matching point x_i, as well as be continuous, that is at each matching point:

(a) $s(x_i) = u_i$, i = 1, 2, ... n, n conditions;

(b) $s_{i+1}(x_i) = s_i(x_i)$, i = 2, 3, ..., n−1, n−2 conditions;

(c) $s'_{i+1}(x_i) = s'_i(x_i)$, i = 2, 3, ..., n−1, n−2 conditions;

(d) $s''_{i+1}(x_i) = s''_i(x_i)$, i = 2, 3, ..., n−1, n−2conditions.

The number of conditions available is thus $4n-6$. Obviously, two extra conditions have to be specified in order to solve this problem. These conditions appear, at this point in our understanding, to be completely arbitrary. For example, these could be in the form of two constraints on the derivatives at the boundaries (these conditions are not to be construed as extra boundary conditions on the differential equation):

(e.1) $s''_1 = u''_1$ and $s''_n = u''_n$ or

(e.2) $s'_1 = u'_1$ and $s'_n = u'_n$.

With conditions (e.1) or (e.2), the spline evaluation problem is closed and can be solved in the same manner as in the previous section. However, a different solving procedure has been applied. The idea here is that in solving Burgers equation, the spline derivatives are of real interest instead of the spline itself. Therefore, a special procedure is set up to

calculate the spline derivatives directly. This follows the same procedure for the cubic spline approximation discussed by Rubin and Graves [16].

This solving procedure consists of two parts which generate two diagonally dominant tridiagonal matrices for the first and second derivatives respectively.
From equation (3.2.1):

$$s_i(x) = a_i + b_i x + c_i x^2 + d_i e^{\lambda_i x} \tag{3.2.2}$$
$$s_i'(x) = b_i + 2c_i x + d_i \lambda_i e^{\lambda_i x} \tag{3.2.3}$$
$$s_i''(x) = 2c_i + d_i \lambda_i^2 e^{\lambda_i x} \tag{3.2.4}$$

First, the subinterval $[x_{i-1}, x_i]$. is considered. Suppose that all the nodal values are denoted as:

$$\begin{aligned} s_i(x_{i-1}) &= u_{i-1} \quad s_i(x_i) = u_i \\ s_i'(x_{i-1}) &= u_{i-1}' \quad s_i'(x_i) = u_i' \\ s_i''(x_{i-1}) &= u_{i-1}'' \quad s_i''(x_i) = u_i'' \end{aligned} \tag{3.2.5}$$

Then two linearly independent systems can be constructed for a_i, b_i, c_i and d_i in terms of node point values defined in (3.2.5):

$$\begin{pmatrix} 1 & x_{i-1} & x_{i-1}^2 & e^{\lambda_i x_{i-1}} \\ 1 & x_i & x_i^2 & e^{\lambda_i x_i} \\ 0 & 1 & 2x_{i-1} & \lambda_i e^{\lambda_i x_{i-1}} \\ 0 & 1 & 2x_i & \lambda_i e^{\lambda_i x_i} \end{pmatrix} \begin{pmatrix} a_i \\ b_i \\ c_i \\ d_i \end{pmatrix} = \begin{pmatrix} u_{i-1} \\ u_i \\ u_{i-1}' \\ u_i' \end{pmatrix} \tag{3.2.6}$$

$$\begin{pmatrix} 1 & x_{i-1} & x_{i-1}^2 & e^{\lambda_i x_{i-1}} \\ 1 & x_i & x_i^2 & e^{\lambda_i x_i} \\ 0 & 0 & 2 & \lambda_i^2 e^{\lambda_i x_{i-1}} \\ 0 & 0 & 2 & \lambda_i^2 e^{\lambda_i x_i} \end{pmatrix} \begin{pmatrix} a_i \\ b_i \\ c_i \\ d_i \end{pmatrix} = \begin{pmatrix} u_{i-1} \\ u_i \\ u_{i-1}'' \\ u_i'' \end{pmatrix} \tag{3.2.7}$$

Similarly, on interval $[x_i, x_{i+1}]$:

$$\begin{aligned} s_{i+1}(x_i) &= u_i \qquad & s_{i+1}(x_{i+1}) &= u_{i+1} \\ s_{i+1}'(x_i) &= u_i' \qquad & s_{i+1}'(x_{i+1}) &= u_{i+1}' \\ s_{i+1}''(x_i) &= u_i'' \qquad & s_{i+1}''(x_{i+1}) &= u_{i+1}'' \end{aligned} \tag{3.2.8}$$

Hence, the two other linearly independent systems are given by:

$$\begin{pmatrix} 1 & x_{i+1} & x_{i+1}^2 & e^{\lambda_{i+1}x_{i+1}} \\ 1 & x_i & x_i^2 & e^{\lambda_{i+1}x_i} \\ 0 & 1 & 2x_{i+1} & \lambda_i e^{\lambda_{i+1}x_{i+1}} \\ 0 & 1 & 2x_i & \lambda_i e^{\lambda_{i+1}x_i} \end{pmatrix} \begin{pmatrix} a_{i+1} \\ b_{i+1} \\ c_{i+1} \\ d_{i+1} \end{pmatrix} = \begin{pmatrix} u_{i+1} \\ u_i \\ u'_{i+1} \\ u'_i \end{pmatrix} \tag{3.2.9}$$

$$\begin{pmatrix} 1 & x_{i+1} & x_{i+1}^2 & e^{\lambda_{i+1}x_{i+1}} \\ 1 & x_i & x_i^2 & e^{\lambda_{i+1}x_i} \\ 0 & 0 & 2 & \lambda_{i+1}^2 e^{\lambda_i x_{i+1}} \\ 0 & 0 & 2 & \lambda_{i+1}^2 e^{\lambda_{i+1}x_i} \end{pmatrix} \begin{pmatrix} a_{i+1} \\ b_{i+1} \\ c_{i+1} \\ d_{i+1} \end{pmatrix} = \begin{pmatrix} u_{i+1} \\ u_i \\ u''_{i+1} \\ u''_i \end{pmatrix} \tag{3.2.10}$$

Solving equation (3.2.6) and (3.2.9) for the coefficients associated with $s_i(x)$ and $s_{i+1}(x)$ at the node conditions: $x_{i-1} = x_i - h$, $x_{i+1} = x_i + h$ gives the coefficients a_i, b_i, and d_i in terms of u_i, u_{i-1}, u'_i and u'_{i-1} and a_{i+1}, b_{i+1}, c_{i+1} and d_{i+1} in terms of u_i, u_{i+1}, u'_i and u'_{i+1}. Substitution of a_i, b_i, c_i, d_i and a_{i+1}, b_{i+1}, c_{i+1}, d_{i+1} back into the equation (3.2.2) gives two functions of x: $s_i(x)$ and $s_{i+1}(x)$. Then, taking the derivative of $s_i(x)$ and $s_{i+1}(x)$ with respect of x twice gives the expressions for $s''_i(x)$ and $s''_{i+1}(x)$. At $x = x_i$, the spline has to satisfy: $s''_i(x) = s''_{i+1}(x)$. This results in:

$$Au'_{i-1} + Bu'_i + Cu'_{i+1} = D(u_i - u_{i-1}) + E(u_i - u_{i-1}) \tag{3.1.11}$$

where:

$$A = \frac{\left[2\left(1-e^{-\lambda_i h}\right)+\left(-2+\lambda_i h\right)\lambda_i h\right]}{h\left[-2\left(1-e^{-\lambda_i h}\right)+\lambda_i h\left(1+e^{-\lambda_i h}\right)\right]}$$

$$B = \frac{\left[-2\left(1-e^{-\lambda_i h}\right)+\left(2e^{-\lambda_i h}+\lambda_i h\right)\lambda_i h\right]}{h\left[-2\left(1-e^{-\lambda_i h}\right)+\lambda_i h\left(1+e^{-\lambda_i h}\right)\right]} - \frac{\left[-2\left(1-e^{\lambda_{i+1} h}\right)+\left(-2e^{\lambda_{i+1} h}+\lambda_{i+1} h\right)\lambda_{i+1} h\right]}{h\left[2\left(1-e^{\lambda_{i+1} h}\right)+\lambda_{i+1} h\left(1+e^{\lambda_{i+1} h}\right)\right]}$$

$$C = \frac{\left[2\left(1-e^{\lambda_{i+1} h}\right)+\left(2e^{\lambda_{i+1} h}+\lambda_{i+1} h\right)\lambda_{i+1} h\right]}{h\left[2\left(1-e^{\lambda_{i+1} h}\right)+\lambda_{i+1} h\left(1+e^{\lambda_{i+1} h}\right)\right]}$$

$$D = \frac{2\lambda_i\left(-1+e^{-\lambda_i h}+\lambda_i h\right)}{h\left[-2\left(1-e^{-\lambda_i h}\right)+\lambda_i h\left(1+e^{-\lambda_i h}\right)\right]}$$

$$E = \frac{2\lambda_{i+1}\left(1-e^{\lambda_{i+1} h}+\lambda_{i+1} h\right)}{h\left[2\left(1-e^{\lambda_{i+1} h}\right)+\lambda_{i+1} h\left(1+e^{\lambda_{i+1} h}\right)\right]}$$

In a similar manner, another set of functions $s_i(x)$ and $s_{i+1}(x)$ can be solved from equations (3.2.7) and (3.2.10). This time, taking the derivative once and matching the first derivative at $x = x_i$ gives an expression for u_i'':

$$au''_{i-1} + bu''_i + cu''_{i+1} = \frac{1}{h}\left(u_{i-1} - 2u_i + u_{i+1}\right) \qquad (3.2.12)$$

where:

$$a = \frac{\left[2\left(1-e^{-\lambda_i h}\right)+(-2+\lambda_i h)\lambda_i h\right]}{2\left(-1+e^{-\lambda_i h}\right)\lambda_i^2 h}$$

$$b = \frac{\left[2\left(1-e^{-\lambda_i h}\right)+\left(-2+\lambda_i h e^{-\lambda_i h}\right)\lambda_i h\right]}{2\left(-1+e^{-\lambda_i h}\right)\lambda_i^2 h} + \frac{\left[2\left(1-e^{\lambda_{i+1} h}\right)+\left(2+\lambda_{i+1} h e^{\lambda_{i+1} h}\right)\lambda_{i+1} h\right]}{2\left(-1+e^{\lambda_{i+1} h}\right)\lambda_{i+1}^2 h}$$

$$c = \frac{\left[2\left(1-e^{\lambda_{i+1} h}\right)+(2+\lambda_{i+1} h)\lambda_{i+1} h\right]}{2\left(-1+e^{\lambda_{i+1} h}\right)\lambda_{i+1}^2 h}$$

With $i \in [2,n-1]$, equations (3.2.10) and (3.2.11) give two diagonally dominant tridiagonal matrices. By solving these two matrices, u_i' and u_i'', the derivatives, are obtained at the node points. Hence the same time marching methods discussed in the previous section can be applied here to find the solution for Burgers equation (3.1).

At the zero velocity points, that is, when $\lambda_i = u_i(x + h/2)/\varepsilon \to 0$, it is found that the coefficients of the tridiagonal systems reduce to constants and equations (3.2.11) and (3.2.12) become:

$$\frac{1}{3}\left(u'_{i-1} + 4u'_i + u'_{i+1}\right) = \frac{1}{h}\left(u_{i+1} - u_{i-1}\right) \qquad (3.2.13)$$

$$\frac{1}{6}(u''_{i-1}+4u''_i+u''_{i+1})=\frac{1}{h^2}(u_{i-1}-2u_i+u_{i+1}) \qquad (3.2.14)$$

equations (3.2.13) and (3.2.14) are identical to the equations that Rubin and Graves [16] derived for the cubic spline interpolation. This is to say that at the zero velocity points, the interpolation functions change to third order polynomials.

To calculate the λ in equation (3.2.1), the mid-point values of $u_i(x_i+h/2)$ or $u_i(x_i-h/2)$ are required. Since two linearly independent expressions for $s_i(x)$ are created, one from the first derivatives and the other one from the second derivatives, there are four expressions for the mid-point values of u:

$$u_i(x_i-h/2)=\frac{1}{2}(u_i+u_{i-1})+ \qquad 3.2.15)$$
$$\frac{\left[\left(4\left(1+e^{-\lambda_i h}-2e^{-\lambda_i h/2}\right)-\lambda_i^2h^2e^{-\lambda_i h}\right)u''_i+\left(4\left(-1-e^{-\lambda_i h}+2e^{-\lambda_i h/2}\right)+\lambda_i^2h^2\right)u''_{i-1}\right]}{8\left(-1+e^{-\lambda_i h}\right)\lambda_i^2}$$

$$u_i(x_i+h/2)=\frac{1}{2}(u_i+u_{i+1})+ \qquad (3.2.16)$$
$$\frac{\left[\left(4\left(1+e^{\lambda_{i+1} h}-2e^{\lambda_{i+1} h/2}\right)-\lambda_{i+1}^2h^2e^{\lambda_{i+1} h}\right)u''_i+\left(4\left(-1-e^{\lambda_{i+1} h}+2e^{\lambda_{i+1} h/2}\right)+\lambda_{i+1}^2h^2\right)u''_{i+1}\right]}{8\left(-1+e^{\lambda_{i+1} h}\right)\lambda_{i+1}^2}$$

$$u_i(x_i-h/2)=\left\{h\left(-1-3e^{-\lambda_i h}+4e^{-\lambda_i h/2}-\lambda_i he^{-\lambda_i h}\right)u'_i+\right.$$
$$h\left(-3-e^{-\lambda_i h}+4e^{-\lambda_i h/2}+\lambda_i h\right)u'_{i-1}+\left[8\left(e^{-\lambda_i h}-e^{-\lambda_i h/2}\right)+\lambda_i h\left(1+3e^{-\lambda_i h}\right)\right]u_i+ \qquad (3.2.17)$$
$$\left.\left[8\left(-1+e^{-\lambda_i h/2}\right)+\lambda_i h\left(3+e^{-\lambda_i h}\right)\right]u_{i-1}\right\}\frac{1}{4\left[2\left(-1+e^{-\lambda_i h}\right)+\left(1+e^{-\lambda_i h}\right)\right]}$$

$$u_i(x_i+h/2)=\left\{h\left(-1-3e^{\lambda_{i+1} h}+4e^{\lambda_{i+1} h/2}+\lambda_{i+1}he^{\lambda_{i+1} h}\right)u'_i+\right. \qquad (3.2.18)$$
$$h\left(-3-e^{\lambda_{i+1} h}+4e^{\lambda_{i+1} h/2}-\lambda_{i+1}h\right)u'_{i+1}+\left[8\left(-e^{\lambda_{i+1} h}+e^{\lambda_{i+1} h/2}\right)+\lambda_{i+1}h\left(1+3e^{\lambda_{i+1} h}\right)\right]u_i+$$
$$\left.\left[8\left(1-e^{\lambda_{i+1} h/2}\right)+\lambda_{i+1}h\left(3+e^{\lambda_{i+1} h}\right)\right]u_{i+1}\right\}\frac{1}{4\left[2\left(1-e^{\lambda_{i+1} h}\right)+\left(1+e^{\lambda_{i+1} h}\right)\right]}$$

In the program, for better accuracy, the mid-point velocity values are updated after each derivative matrix call, that is, they are computed once after the second derivative matrix is called with the equations (3.2.15) and (3.2.16) and then, the new values of λ_i are used to call the first derivative matrix, once the first derivatives are available, the λ_i are computed again from equations (3.2.17) and (3.2.18). The mid-point values of velocity are computed from the equations (3.2.16) and (3.2.18) for the left boundary while equations (3.2.15) and (3.2.17) are used for all the other node points. The algorithm is completed with a time marching process of the form (3.1.11) above.

CHAPTER 4

THE LAMB VORTEX PROBLEM

The exact solution to the Navier-Stokes equations that generates the so-called Lamb vortex is a good example to test the accuracy of numerical methods since diffusion is a dominant feature of the flow and must be correctly estimated to get a meaningful simulation of the flow field.

4.1 Description of the problem

The Lamb vortex is an exact solution [18] of the unsteady, incompressible Navier-Stokes equations (2.1.10). The circumferential velocity is given by:

$$V_\theta = \Gamma_0\left(1 - e^{-(r^2/4\varepsilon t)}\right)/2\pi r \tag{4.1.1}$$

where Γ_0 is the vortex strength. The radial velocity component is zero and the pressure is given by the quadrature:

$$p(r,t) = p_{core} + \int_0^r \frac{\rho V_\theta^2}{r'} dr' \tag{4.1.2}$$

The purpose in choosing the Lamb vortex is to examine the dissipative nature of the generalized Roscoe method. As Burgers' equation results suggest, observation of diminished slopes in steep gradient regions is an indication of spatial dissipation. The Lamb vortex presents the opportunity for the spatial dissipation to accumulate in time to give a progressively worse solution. Here a comparison among the two generalized Roscoe procedures, the central difference method of Abdallah [19] with second order accuracy and a cubic spline scheme is presented.

The flow conditions for the problem are $\Gamma_0 = 10\,ft^2/sec$, $\varepsilon = 0.025\,ft^2/sec$ is the kinematic viscosity, $\rho = 0.002378\,slug/ft^3$, and $p_{core} = 100\,lb_f/ft^2$. The grid uses 100×100 points on the domain $x = [-5,5]$ and $y = [-5,5]$. This gives a peak Reynolds number of about 400 and an average cell Reynolds number of 4. For the stated initial conditions, the exact Cartesian velocities are computed at $t = 5\,sec$ for a vortex centered at (0,0). The initial pressure was computed from numerical integration of the pressure integral. The solution is advanced in time to 12.28 seconds.

Four numerical algorithms are devised. These are Abdallah's central differencing of the unsteady incompressible Navier-Stokes equations [19], the cubic spline scheme derived from the Rubin and Graves [16] one dimensional formulation, and the linear and quadratic exponential spline methods developed directly from the generalized Roscoe differencing. For all four algorithms, a backward Euler time marching procedure is used for the momentum equations, and an Approximate-Factorization (AF) scheme for the pressure Poisson equation. The boundaries are not really deemed to affect the solution near the core, at least for the small times shown, since these are relatively far from the core. This is borne out in the computations.

4.2 Numerical procedure for the linear exponential spline method

The unsteady, incompressible Navier-Stokes equations (2.1.10) are repeated here for reference:

$$\begin{aligned}
&\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0 \\
&\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}=\frac{1}{Re}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)-\frac{\partial p}{\partial x} \\
&\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}=\frac{1}{Re}\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right)-\frac{\partial p}{\partial y}
\end{aligned} \tag{4.2.1}$$

By taking the divergence of the momentum equations and forcing the resulting equation to satisfy the continuity equation, the Poisson equation for pressure is obtained:

$$\frac{\partial^2 p}{\partial^2 x}+\frac{\partial^2 p}{\partial^2 y}=-\left(\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial v}{\partial y}\right)^2+2\frac{\partial u}{\partial y}\frac{\partial v}{\partial x}\right) \tag{4.2.2}$$

In order to solve the system (4.2.1) numerically, a two-part solving procedure is used. One part is in the spatial domain and the other is in the time domain. At each time level, the derivatives with respect to time are taken as constants while considering the spatial discretization. First, the generalized Roscoe scheme is applied in each spatial direction to obtain two systems related to the system (4.2.1), one in the x direction and the other one in the y direction:

$$\frac{\partial u}{\partial x} = r_1$$

$$U\frac{\partial u}{\partial x} - \frac{1}{Re}\frac{\partial^2 u}{\partial x^2} + \frac{\partial p}{\partial x} = r_2 \qquad (4.2.3)$$

$$U\frac{\partial v}{\partial x} - \frac{1}{Re}\frac{\partial^2 v}{\partial x^2} = r_3$$

$$\frac{\partial v}{\partial y} = R_1$$

$$V\frac{\partial u}{\partial y} + \frac{1}{Re}\frac{\partial^2 u}{\partial y^2} = R_2 \qquad (4.2.4)$$

$$V\frac{\partial v}{\partial y} + \frac{1}{Re}\frac{\partial^2 v}{\partial y^2} + \frac{\partial p}{\partial y} = R_3$$

When equation (4.2.3) is built, the variation in the y direction is neglected, thus all y direction derivative terms are taken as constants and put into the Roscoe vector $\vec{r}$ on the right hand side. The same treatment is applied for the x direction derivatives in equation (4.2.4). If the forcing functions of $\vec{r}^T = (r_1, r_2, r_3)$ and $\vec{R}^T = (R_1, R_2, R_3)$ are defined as in equations (2.2.22) and (2.2.23), the velocity components u, v and pressure p have the structure of a linear exponential function of x as in equation (2.2.24). System (4.2.4) gives a similar structure for the y-dependence of the dependent variables. This gives the functions that are used in the spline interpolations. Unlike the one dimensional Burgers equation where there was only one unknown variable in one direction, there are now three unknown variables u, v and p to be interpolated in the two direction, x and y. The six interpolating functions are:

$$\begin{aligned}
u(x, y_n) &= a_{x_u} + b_{x_u} x + c_{x_u} e^{\lambda_x x} \\
v(x, y_n) &= a_{x_v} + b_{x_v} x + c_{x_v} e^{\lambda_x x} \\
p(x, y_n) &= a_{x_p} + b_{x_p} x + c_{x_p} e^{\lambda_x x} \\
u(x_n, y) &= a_{y_u} + b_{y_u} y + c_{y_u} e^{\lambda_y y} \\
v(x_n, y) &= a_{y_v} + b_{y_v} y + c_{y_v} e^{\lambda_y y} \\
p(x_n, y) &= a_{y_p} + b_{y_p} y + c_{y_p} e^{\lambda_y y}
\end{aligned} \qquad (4.2.5)$$

where $\lambda_x = U/\varepsilon$, $\lambda_y = V/\varepsilon$. The x or y subscript stands for the direction on which the interpolation is carried out while the second subscripts u, v and p for the dependent variable to be interpolated. The connection between the velocity components or the connection between velocity components and pressure is thus established via the parameters λ_x and λ_y during the spline interpolation.

After the interpolation, the derivatives $\frac{\partial u}{\partial x}$, $\frac{\partial^2 u}{\partial x^2}$, $\frac{\partial v}{\partial x}$, $\frac{\partial^2 v}{\partial x^2}$, $\frac{\partial p}{\partial x}$ and $\frac{\partial u}{\partial y}$, $\frac{\partial^2 u}{\partial y^2}$, $\frac{\partial v}{\partial y}$, $\frac{\partial^2 v}{\partial y^2}$, $\frac{\partial p}{\partial y}$ are calculated directly from the coefficients according to equations (3.1.9) and (3.1.10). Then the forcing function in the pressure Poisson equation is computed according to equation (4.2.2). The pressure Poisson equation is then solved in an iterative fashion. Thus, all the spatial derivatives and pressure values at every discrete point (i, j) are available. The next step is to solve the time derivatives by time marching the momentum equations with all the calculated velocity and pressure derivatives.

4.3. Numerical procedure for the quadratic exponential spline method

In this section, it is assumed that all the dependent variables in system (4.2.3) and (4.2.4) behave like quadratic exponential functions (it should be noted here that this assumption is not a result from the analytical study of the two dimension incompressible system (4.2.1) like in the linear exponential case, it is purely an extension from the one dimension system result). The interpolating functions are now given, in the similar notation as in equation (4.2.5), by:

$$
\begin{aligned}
u(x,y_n) &= a_{x_u} + b_{x_u}x + c_{x_u}x^2 + d_{x_u}e^{\lambda_x x} \\
v(x,y_n) &= a_{x_v} + b_{x_v}x + c_{x_v}x^2 + d_{x_v}e^{\lambda_x x} \\
p(x,y_n) &= a_{x_p} + b_{x_p}x + c_{x_p}x^2 + d_{x_p}e^{\lambda_x x} \\
u(x_n,y) &= a_{y_u} + b_{y_u}y + c_{y_u}y^2 + d_{y_u}e^{\lambda_y y} \\
v(x_n,y) &= a_{y_v} + b_{y_u}y + c_{y_u}y^2 + d_{y_u}e^{\lambda_y y} \\
p(x_n,y) &= a_{y_p} + b_{y_p}y + c_{y_p}y^2 + d_{y_p}e^{\lambda_y y}
\end{aligned}
\qquad (4.3.1)
$$

and again $\lambda_x = U/\varepsilon$, $\lambda_y = V/\varepsilon$.

The numerical solving procedure is virtually the same as in the previous section except that, like in the one dimensional case, instead of solving for the coefficients of the

spline functions the derivatives are solved directly from matrices derived from the spline interpolation. This time, two special two-dimensional matrix solvers are devised, one is used for solving for the first derivatives while the other one is for obtaining for the second derivatives.

4.4 Numerical procedure for the cubic spline method

The numerical procedure for the cubic spline algorithm derived from equation (3.2.12) and (3.2.13) is very similar in its structure to that of quadratic exponential spline. Since the matrices based on equation (3.2.12) and (3.2.13) are constant matrices, the set up is much simpler.

CHAPTER 5

STABILITY AND ACCURACY STUDY

Stability and accuracy of a numerical scheme are two fundamental properties which need to be determined in order to compare with other schemes. In this chapter, the stability and accuracy of the two exponential spline schemes are studied.

5.1 Accuracy study

To find the spatial order of accuracy of the generalized Roscoe methods and to compare this accuracy with other methods, a series of computations with different grid spacings were performed . Two problems were examined, the steady nonlinear Burgers equation, presented in Figures 5.1.1-5.1.11, and unsteady solutions to the incompressible Navier-Stokes equations, set up for the Lamb vortex problem shown in Fig. 5.1.12-5.1.15. In both of the problems examined, physical and phase plane solutions obtained from several methods are presented including the two generalized Roscoe methods. An error analysis is performed to assess the order of accuracy of the generalized Roscoe schemes.

A series of computations for the nonlinear Burgers equation were performed for cell Reynolds numbers of 0.625, 1.25, 2.5, and 5.0. The numerical solutions are compared with the exact solution in Figures 5.1.1-5.1.6. For ease of comparison, all the Roscoe-based methods use filled in symbols.

At the lower cell Reynolds numbers, none of the schemes have any difficulty in achieving an accurate solution. This is seen in Figures 5.1.1 and 5.1.2 where all methods converge easily and are monotonic. As the cell Reynolds number is increased beyond two, however, the central difference, fourth-order compact, and cubic spline methods begin to violate the maximum principle possessing an oscillatory solution and smaller time steps are required to achieve a stable solution. Eventually, there is no time step small enough for convergence and results for those methods can no longer be presented. Figure 5.1.3 clearly shows this behavior for the central difference method. The fourth-order compact scheme is also oscillatory in this figure, but to a much smaller extent and this is not obvious in the figure. Neither of the two methods were stable for a cell Reynolds number of 5 and only the cubic spline remains of these methods in Figure 5.1.4. This leaves only the Roscoe-based schemes which are stable for higher cell Reynolds

numbers as shown in Figures 5.1.5 and 5.1.6. The solutions remain monotonic and appear to retain adequate accuracy.

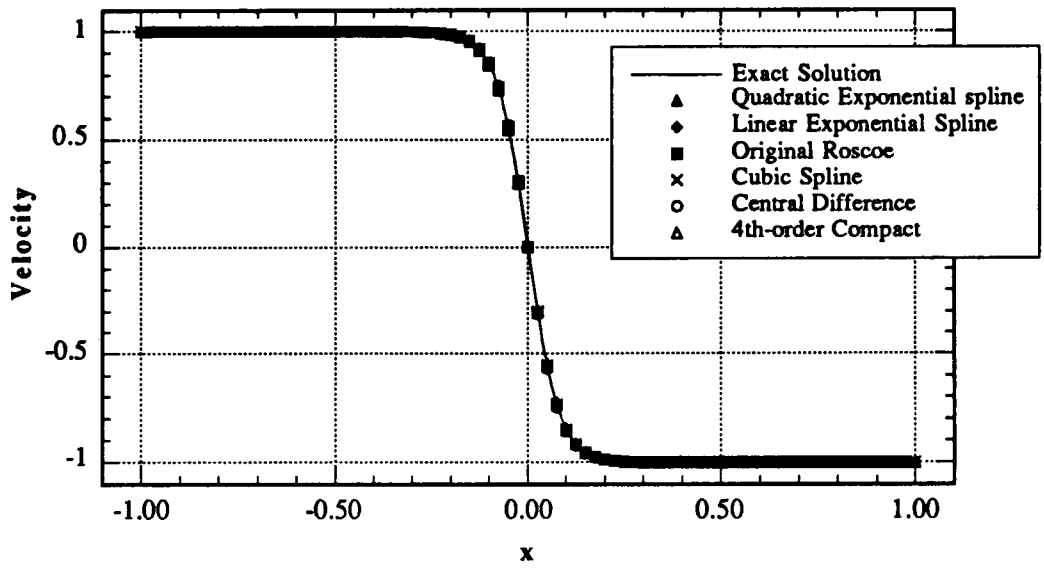


Figure 5.1.1 Numerical Solutions for Burgers Equation. Re_{cell}=0.625.

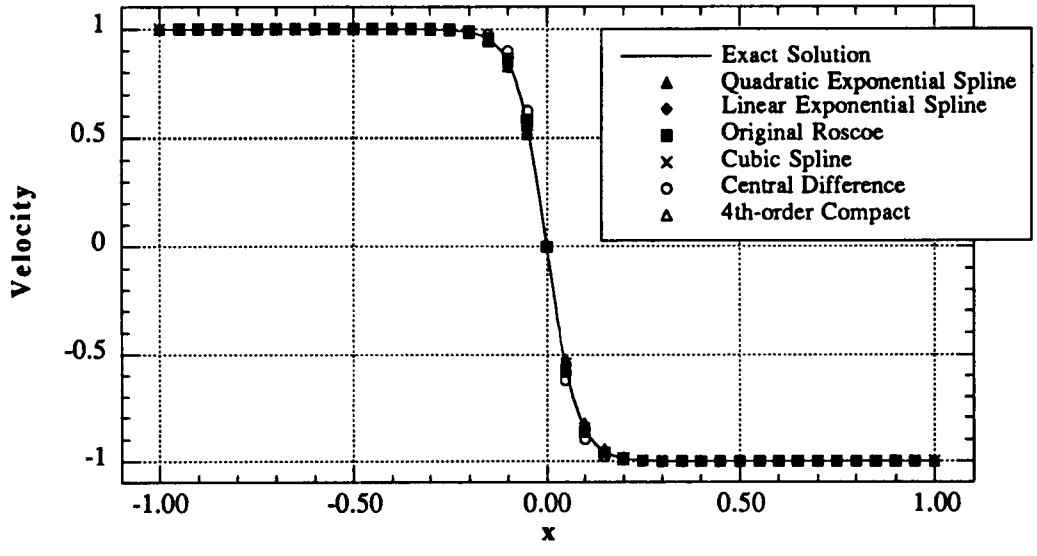


Figure 5.1.2 Numerical Solutions for Burgers Equation. Re_{cell} =1.25

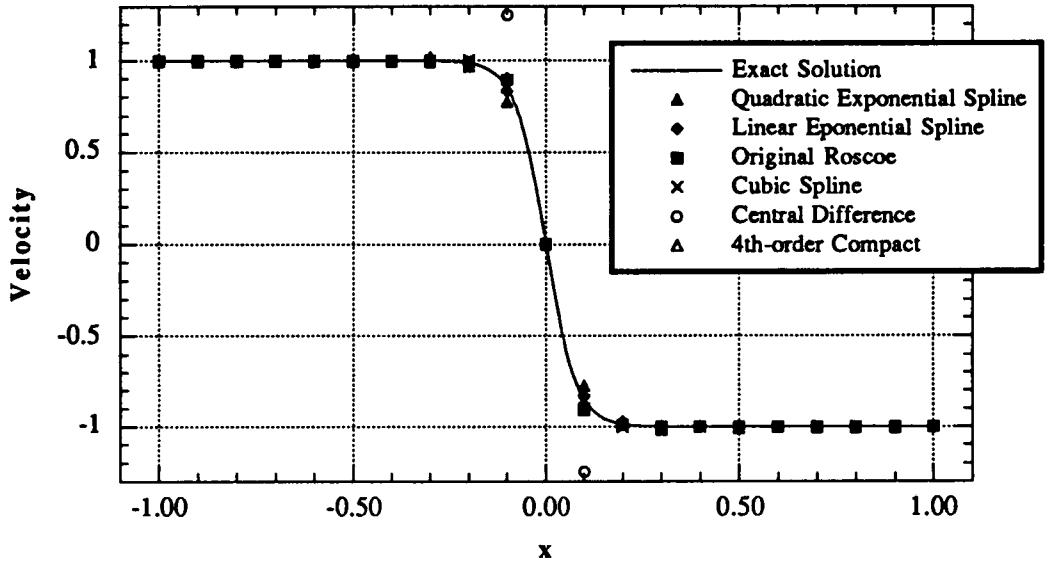


Figure 5.1.3 Numerical Solutions for Burgers Equation. Re_{cell} =2.5

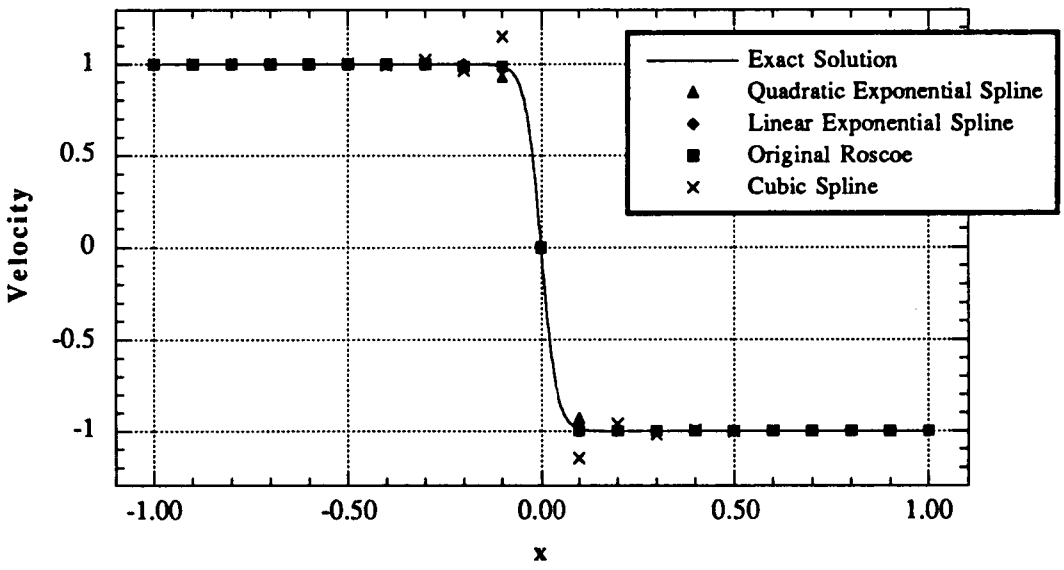


Figure 5.1.4 Numerical Solutions for Burgers Equation. Re_{cell} =5.0

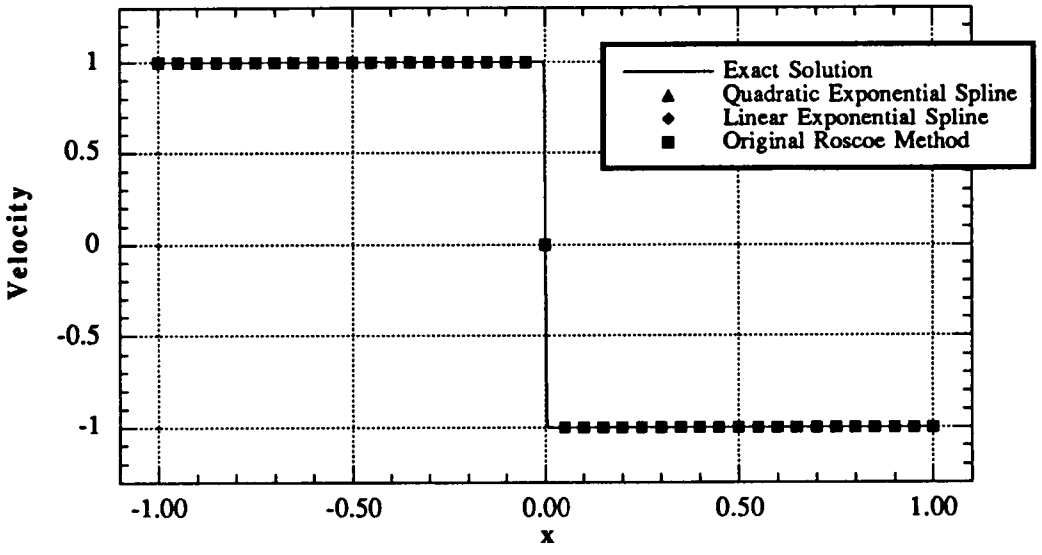


Figure 5.1.5 Numerical Solutions for Burgers Equation. Re_{cell} =50

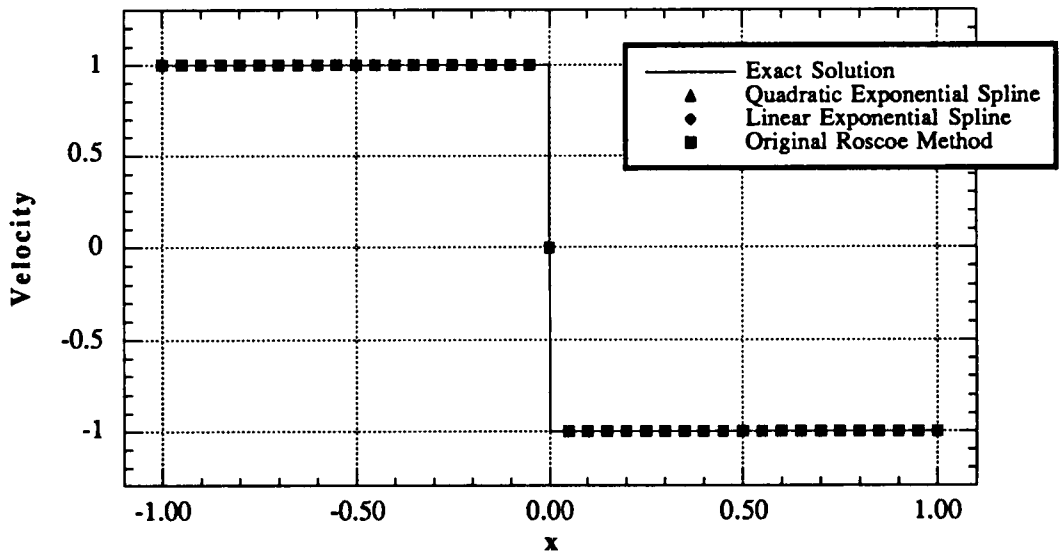


Figure 5.1.6 Numerical Solutions for Burgers Equation. Re_{cell} =500

Phase space solutions (velocity versus velocity derivative) for cell Reynolds numbers of .625, 1.25, 2.5, and 5.0 are shown in Figures 5.1.7-5.1.10. Again the Roscoe based methods use a filled in symbol for ease of identification. The methods which present solutions closest to the curve are the more accurate, of course. An interesting observation of the phase plane studies was that the methods which remain on the inside of the phase plane exact solution tended to remain stable while those which presented solutions outside the exact solution curve eventually became unstable. In this regard, the Roscoe based methods all remained inside the exact solution while the other methods all started on the outside and eventually diverged as the cell Reynolds number was increased.

For a cell Reynolds number of .625, shown in Figure 5.1.7, all of the methods yield solutions which are fairly close to the phase space solution. While it is somewhat difficult to distinguish the symbols, it is evident even in this figure that the Roscoe-based methods all lie either on the phase space curve or to the inside of the curve, while the others are slightly outside. This is indicative of later stability difficulties.

In Figure 5.1.8, the cell Reynolds number is 1.25 and the central difference method gives solutions which are furthest from the curve. The largest error of any of the methods is at the vertical axis which corresponds to the center of the shock location. It is interesting that the central difference method gives a rather good slope for this cell Reynolds number.

As the cell Reynolds number is increased to 2.5, shown in Figure 5.1.9, both the fourth-order compact and central difference methods yield points which are on the positive side of the horizontal phase plane axis near the ends of the curve. This violates the maximum principle and indicates that these two methods are about to go unstable. The cubic spline routine now also displays some points outside the curve at this cell Reynolds number. The Roscoe based methods remain within the exact solution curve.

At a cell Reynolds number of 5, Figure 5.1.10, only the Roscoe methods remain along with the cubic spline method which now shows a slight violation of the maximum principle at the ends of the curve. The Roscoe methods are again on the inside.

Figure 5.1.11 presents the L_2 norm of the error for a given method versus the grid size. Since the graph is presented on a log-log scale, the slope of the line equals the order of accuracy of the method. Since the vertical axis is the L_2 norm of the error, the lower a line is, the more accurate is the solution. The comparison of the numerical results shows clearly that the two exponential spline schemes are second order with respect to the spatial variable and the linear exponential spline scheme displays the greatest accuracy compared with all the other second order schemes. The fourth order compact scheme is

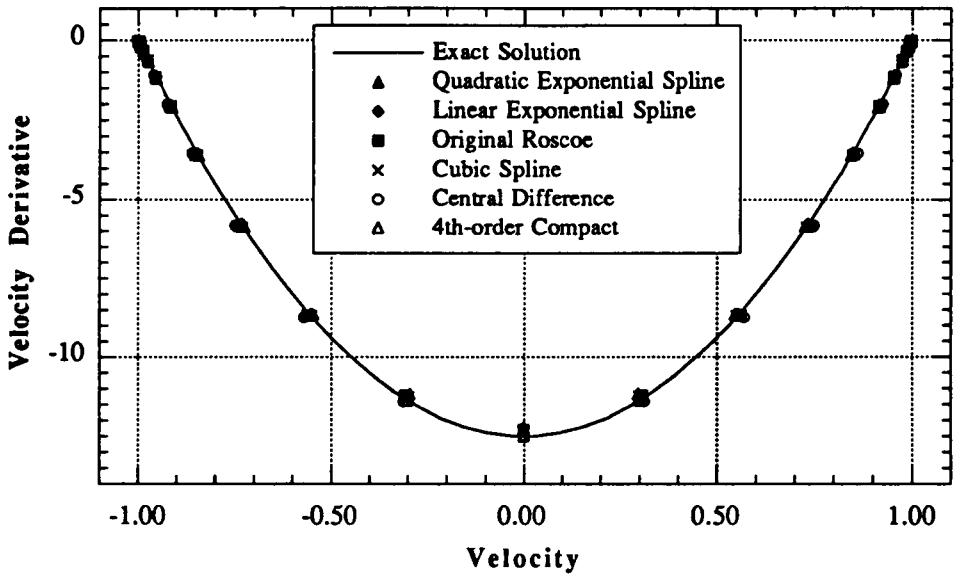


Figure 5.1.7 Phase Space Solutions for Burgers Equation. Re_{cell} =0.625

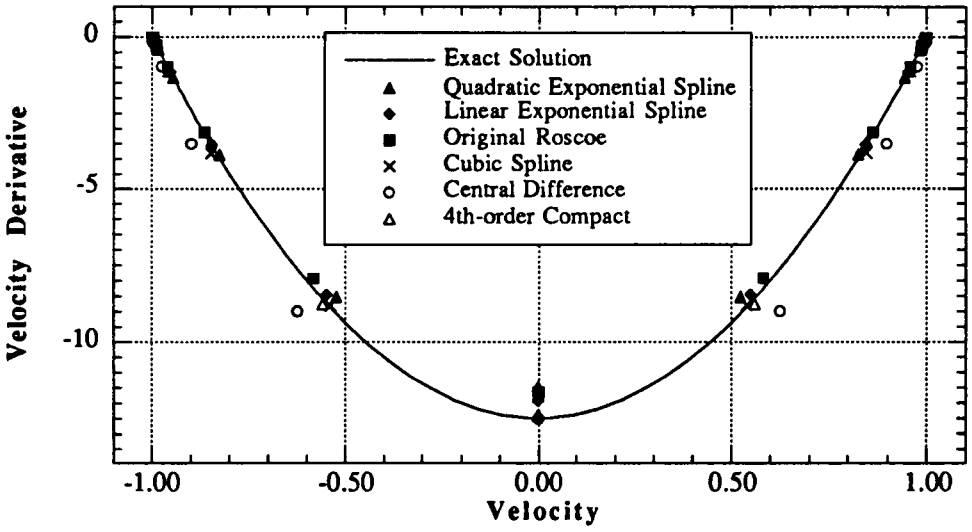


Figure 5.1.8 Phase Space Solutions for Burgers Equation. Re_{cell} =1.25

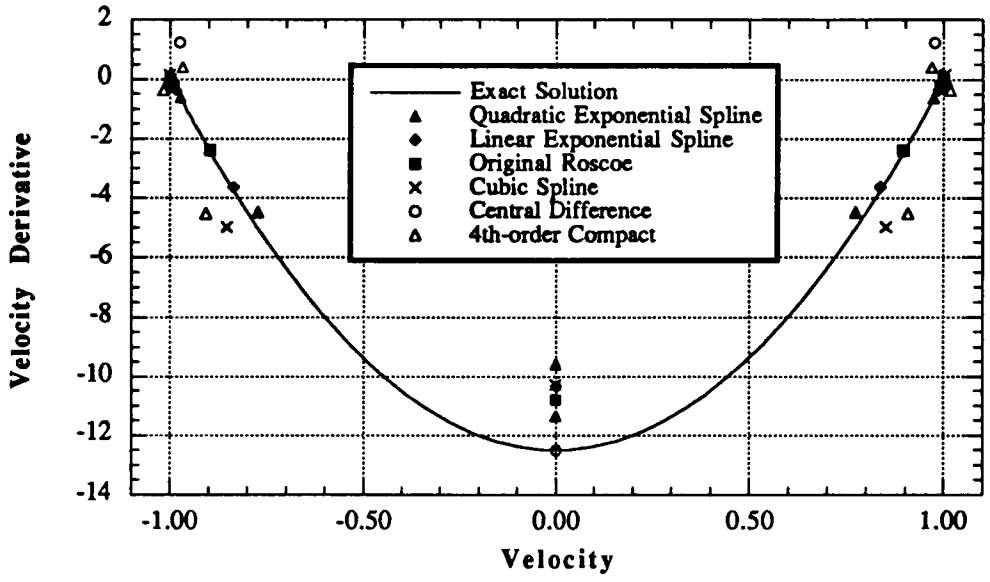


Figure 5.1.9 Phase Space Solutions for Burgers Equation. Re_{cell} =2.5

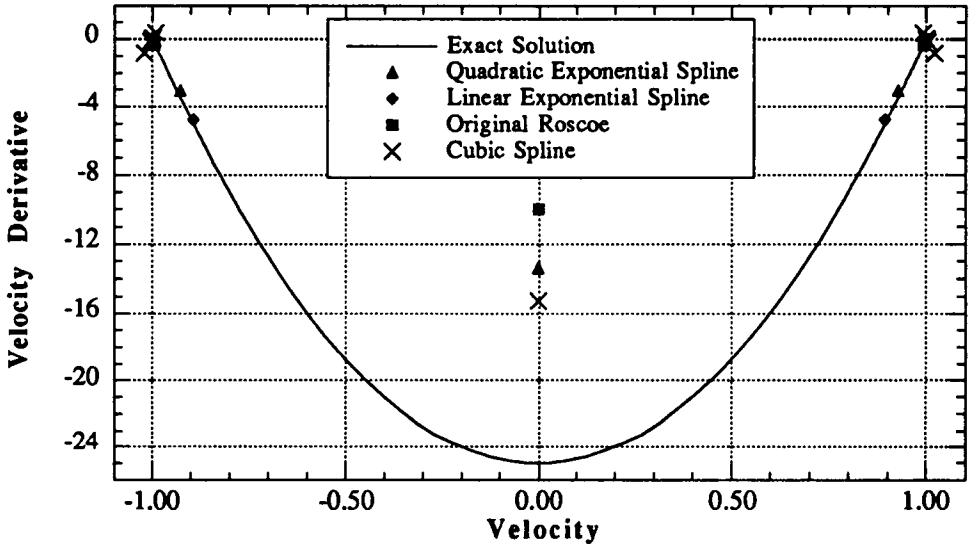


Figure 5.1.10 Phase Space Solutions for Burgers Equation. Re_{cell} =5.0

the most accurate for the smaller values of dx, but becomes less accurate than the three spline schemes as dx approaches to 0.1 which corresponds to a cell Reynolds number of about 2. For higher cell Reynolds numbers, only the first order accurate upwind method and the two Roscoe spline methods remained stable without the explicit addition of artificial viscosity terms. Since computations of flow problems on practical grids typically have cell Reynolds numbers much higher than 2, this indicates the superiority of the Roscoe spline methods.

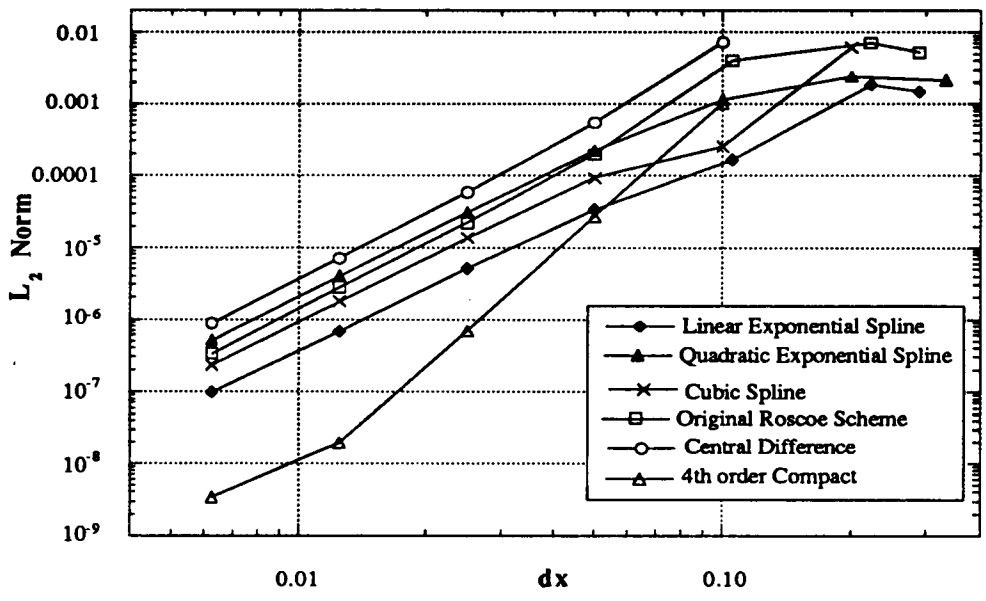


Fig. 5.1.11. Accuracy Comparison of the Numerical Solutions of Burger Equation.

The second problem studied was the Lamb vortex. Here the unsteady incompressible Navier-Stokes equations were solved on a rectangular grid with the vortex center at the center of the grid. The singularity is avoided by having the exact center of the vortex located at a grid cell center. The solutions were marched in time from the exact initial conditions at t = 5 to a time of t = 12.28. The boundaries were sufficiently far away that their influence was negligible.

The solutions for three different grids are shown in Figures 5.1.12-5.1.14. The two Roscoe based methods again use the filled in symbols. The figures show the circumferential velocity versus the distance from the core. Two radii are given for each method with the points being equidistant along each radii. Thus if there is no discernible

difference between the two solutions, the method has maintained the symmetry of the solution.

Sources of error include the time discretization and spatial dissipation. Since all methods used the same time differencing, differences in time accuracy errors between the methods is believed to be slight. Thus, differences in the solutions are more likely to be due to spatial dissipation errors.

Figure 5.1.12 presents the circumferential velocity solutions for a grid of 100x100 points. The best solution is given by the quadratic exponential method with the linear exponential and cubic spline methods giving slightly less accurate results. The central difference method gives a significantly less accurate solution. Near the core, the central difference method shows double symbols indicating the beginnings of a lack of symmetry in the solution. The other methods remain symmetric.

These same trends are followed but to a more exaggerated extent in Figures 5.1.13 and 5.1.14 with grids of 80x80 points and 60x60 points respectively. The quadratic exponential spline method retains the best accuracy throughout with the central difference method remaining the least accurate.

Comparisons of the solutions on the different grids with the exact solution were used to create an error versus grid size chart shown in Figure 5.1.15. The L_2 norm was used as the error estimate. The figure shows that the quadratic exponential spline method is superior over the linear exponential spline, the cubic spline and of course the central difference methods at coarse and medium grid. For a much finer grid, the quadratic spline accuracy appears to be less than the cubic spline method. The reasons for this are not known.

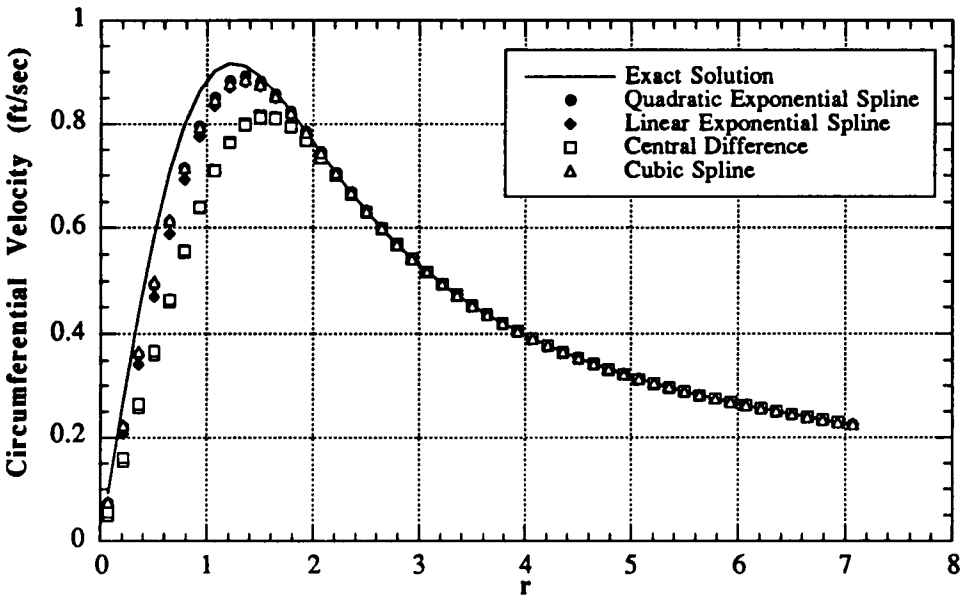


Figure 5.1.12 Numerical Solutions for the Lamb Vortex on the 100x100 grid at t= 12.28 sec.

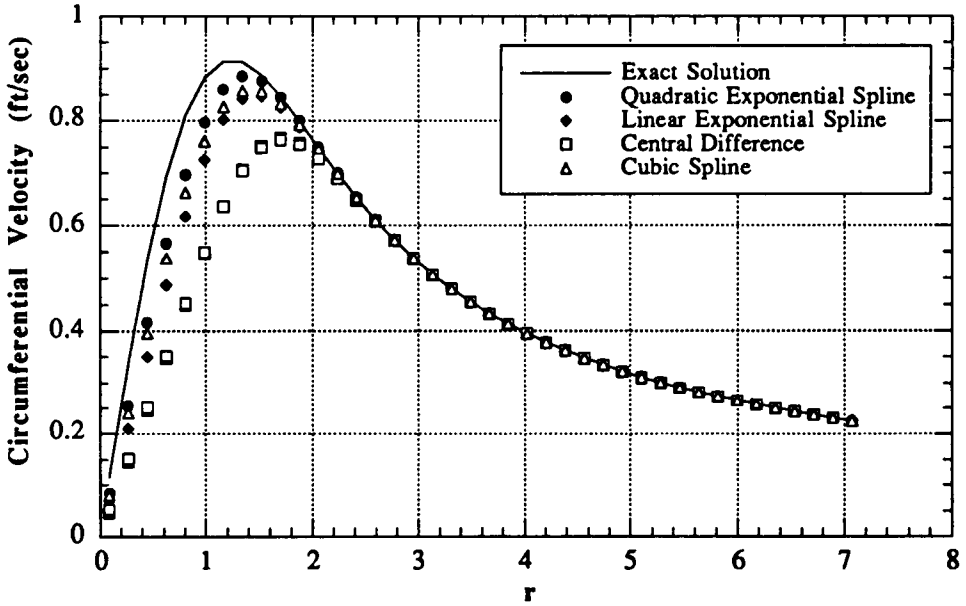


Figure 5.1.13 Numerical Solutions for the Lamb Vortex on the 80x80 grid at t= 12.28 sec.

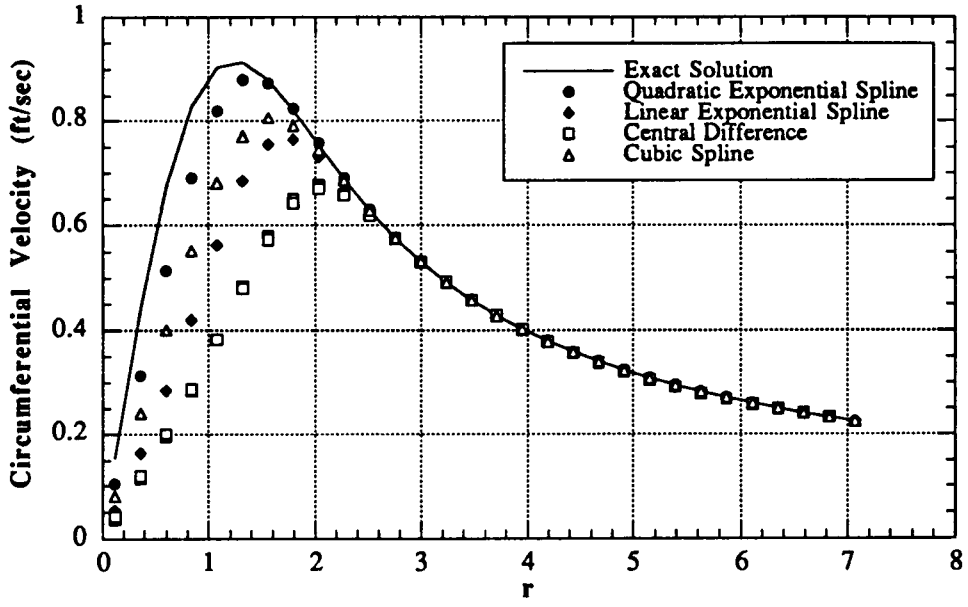


Figure 5.1.14 Numerical Solutions for the Lamb Vortex on the 60x60 grid at t= 12.28 sec.

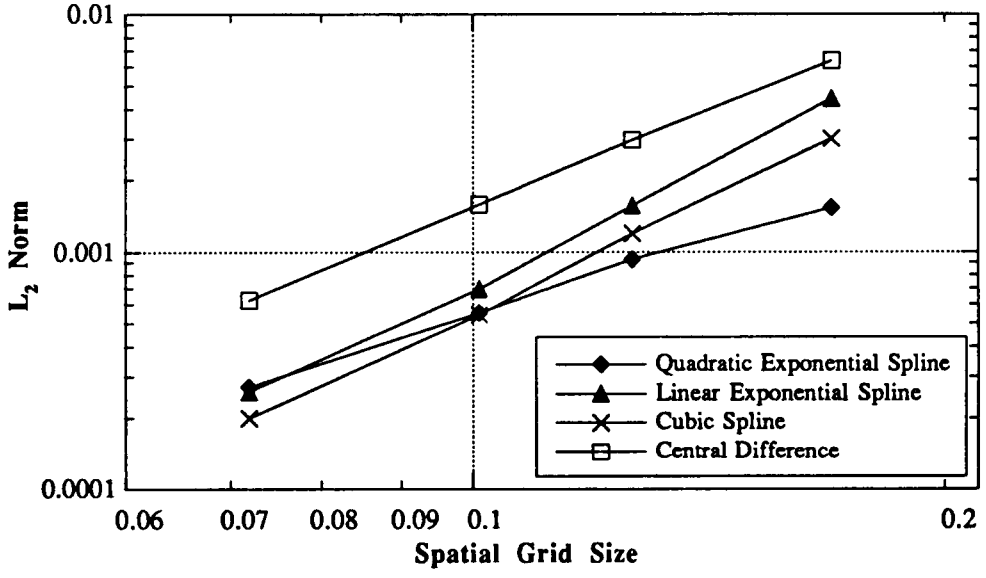


Figure 5.1.15 Accuracy Comparison of the Numerical Solutions to the Lamb Vortex.

5.1.1 Accuracy study for the linear exponential spline scheme

In Chapter 3, the application of the linear exponential spline method was discussed in detail along with the example problem of the steady state Burgers equation. Generally, the process includes three steps:

(i) Solving the coefficients of the spline interpolating functions throughout the calculation domain so that the overall spline function is known.

(ii) Differentiating the spline function to get the derivatives at the node points.

(iii) Solving the discretized differential equation by time marching.

The overall accuracy is determined by the accuracy of the numerical techniques adapted in each of these three steps. In step (ii), the analytical differentiation is made on the spline interpolation function itself, thus, there is no additional discretization error generated in this step. In the time marching step, a backward Euler method is applied that is known to be of first order accuracy with respect to time. Thus, the accuracy for this scheme is mainly determined by step (i), the spline interpolating process. The accuracy of this step includes:

(i) accuracy of the interpolating function, that is, how close the local interpolating function represents the exact solution. For this method, Roscoe [4] has proved that the linear exponential interpolating function is of second order for viscous flow and first order for inviscid flow.

(ii) accuracy of the interpolating procedure. The current effort is concentrated here.

The best approximation theorem in generalized spline theory [15] indicates that the best approximation result can be expected from a global spline interpolation with a chosen interpolation function. However, there is no error estimation theorem available for the exponential splines. But, from the best approximation theorem, it is certain that the global approximation accuracy is no lower than that of the local interpolation function. Thus, the accuracy study may start with the local spline error estimation.

In Chapter 3, the linear exponential spline interpolants were given as:

$$u(x)_i = c_{1_i} + \left(\frac{r_i}{U_i}\right)x + c_{2_i}e^{\lambda_i x}$$
$$\lambda_i = \frac{U_i}{\varepsilon} \qquad (5.1.1)$$

where r_i/U_i is taken to be another unknown coefficient in the general definition 2.1.1. The defined domain for each spline function, $s_i(x)$, is $[x_i - h/2, x_i + h/2]$ for the interior points and $[x_2 - h, x_2 + h/2]$, $[x_{n-1} - h/2, x_{n-1} + h]$ for the left and right boundary

respectively. In addition, the spline interpolants match the value and first derivative at the middle points on each grid interval. The splines are also required to pass through the grid nodal points, x_i, which are the middle points of the defined domain for each spline interpolant. This gives the same number of conditions as unknowns for a linear system. Hence the spline can be determined without adding any extra constraints which are not naturally part of the problem specification. In the accuracy study, however, this is not the concern. The objective here is to find a discretized algebraic expression which satisfies the spline interpolating requirements. Considering a segment of the spline shown in Figure 5.1.16 with three connected subdomains m, i, p defined on the intervals $[x_{i-1}-h/2, x_{i-1}+h/2]$, $[x_i-h/2, x_i+h/2]$ and $[x_{i+1}-h/2, x_{i+1}+h/2]$ respectively: The corresponding interpolation functions are defined accordingly:

$$u(x)_m = c_{1m} + \left(\frac{r_{i-1}}{U_{i-1}}\right)x + c_{2m}e^{\lambda_{i-1}x} \qquad (5.1.2)$$

$$\lambda_{i-1} = \frac{U_{i-1}}{\varepsilon}$$

$$u(x)_i = c_{1i} + \left(\frac{r_i}{U_i}\right)x + c_{2i}e^{\lambda_i x} \qquad (5.1.3)$$

$$\lambda_i = \frac{U_i}{\varepsilon}$$

and

$$u(x)_p = c_{1p} + \left(\frac{r_{i+1}}{U_{i+1}}\right)x + c_{2p}e^{\lambda_{i+1}x} \qquad (5.1.4)$$

$$\lambda_{i+1} = \frac{U_{i+1}}{\varepsilon}$$

with a globally constant diffusion constant ε.

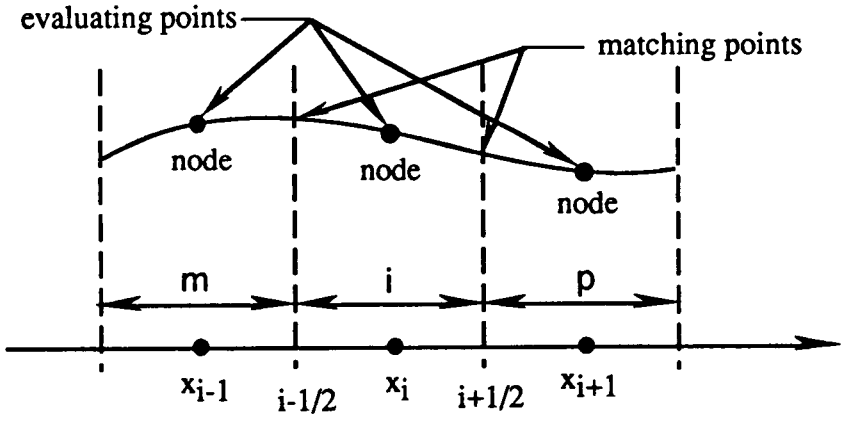


Figure 5.1.16 Interior of the Linear Exponential Spline

By requiring that:

(i) The interpolating functions $u(x)_m$, $u(x)_i$ and $u(x)_p$ must pass through the node points x_{i-1}, x_i and x_{i+1}.

(ii) $u(x)_m$, $u(x)_i$ and $u(x)_i$, $u(x)_p$ must match values at the middle points of the intervals $[x_{i-1},x_i]$ and $[x_i,x_{i+1}]$ respectively.

(iii) $u(x)_m$, $u(x)_i$ and $u(x)_i$, $u(x)_p$ must match first derivative at the middle points of the interval $[x_{i-1},x_i]$ and $[x_i,x_{i+1}]$ respectively.

The coefficients c_{1_m}, c_{1_i} and c_{1_p} can be eliminated and the following relations derived:

$$\begin{aligned}
&u_{i-1} - c_{2_m} + \frac{r_{i-1}}{U_{i-1}}\left(\frac{h}{2}\right) + c_{2_m}e^{\lambda_{i-1}\frac{h}{2}} = u_i - c_{2_i} - \frac{r_i}{U_i}\left(\frac{h}{2}\right) + c_{2_i}e^{-\lambda_i\frac{h}{2}} \\
&\frac{r_{i-1}}{U_{i-1}} + c_{2_m}\lambda_{i-1}e^{\lambda_{i-1}\frac{h}{2}} = \frac{r_i}{U_i} + c_{2_i}\lambda_i e^{-\lambda_i\frac{h}{2}} \\
&u_i - c_{2_i} + \frac{r_i}{U_i}\left(\frac{h}{2}\right) + c_{2_i}e^{\lambda_i\frac{h}{2}} = u_{i+1} - c_{2_p} - \frac{r_{i+1}}{U_{i+1}}\left(\frac{h}{2}\right) + c_{2_p}e^{-\lambda_{i+1}\frac{h}{2}} \\
&\frac{r_i}{U_i} + c_{2_i}\lambda_i e^{\lambda_i\frac{h}{2}} = \frac{r_{i+1}}{U_{i+1}} + c_{2_p}\lambda_{i+1}e^{-\lambda_{i+1}\frac{h}{2}}
\end{aligned} \tag{5.1.5}$$

with λ_{i-1}, λ_i and λ_{i+1} defined as in equations (5.1.2), (5.1.3) and (5.1.4). Elimination of the three unknown coefficients c_{2_m}, c_{2_i} and c_{2_p} gives a three point stencil expression:

$$a_m u_{i-1} - a_i u_i + a_p u_{i+1} = b_m\frac{r_m}{U_m} + b_i\frac{r_i}{U_i} + b_p\frac{r_p}{U_p} \tag{5.1.6}$$

where:

$$a_m = \left[\lambda_{i-1}\lambda_i\left(1 - e^{\lambda_{i+1}\frac{h}{2}}\right)e^{(2\lambda_i+\lambda_{i-1})\frac{h}{2}} + \lambda_{i-1}\lambda_{i+1}\left(1 - e^{\lambda_i\frac{h}{2}}\right)e^{(\lambda_{i-1}+\lambda_i)\frac{h}{2}}\right]$$

$$a_i = \left[\lambda_{i+1}\lambda_i\left(1 - e^{\lambda_{i-1}\frac{h}{2}}\right) + \lambda_i\lambda_{i+1}\left(1 - e^{\lambda_i h}\right)e^{\lambda_{i-1}\frac{h}{2}} + \lambda_{i-1}\lambda_i\left(1 - e^{\lambda_{i+1}\frac{h}{2}}\right)e^{(2\lambda_i+\lambda_{i-1})\frac{h}{2}}\right]$$

$$a_p = \left[\lambda_i\lambda_{i+1}\left(1 - e^{\lambda_{i-1}\frac{h}{2}}\right) + \lambda_{i-1}\lambda_{i+1}\left(1 - e^{\lambda_i\frac{h}{2}}\right)e^{\lambda_{i-1}\frac{h}{2}}\right]$$

and

$$b_m = \lambda_i\left(1-e^{\lambda_{i-1}\frac{h}{2}}\right)\left(1-e^{\lambda_{i+1}\frac{h}{2}}\right)e^{\lambda_i h} + \lambda_{i+1}\left(1-e^{\lambda_{i-1}\frac{h}{2}}\right)\left(1-e^{\lambda_i\frac{h}{2}}\right)e^{\lambda_i\frac{h}{2}}$$
$$+\lambda_{i-1}\lambda_i\frac{h}{2}\left(1-e^{\lambda_{i+1}\frac{h}{2}}\right)e^{(\lambda_{i-1}+2\lambda_i)\frac{h}{2}} + \lambda_{i-1}\lambda_{i+1}\frac{h}{2}\left(1-e^{\lambda_i\frac{h}{2}}\right)e^{(\lambda_{i-1}+\lambda_i)\frac{h}{2}}$$

$$b_i = -\lambda_i\left(1-e^{\lambda_{i-1}\frac{h}{2}}\right)\left(1-e^{\lambda_{i+1}\frac{h}{2}}\right)\left(1-e^{\lambda_i h}\right) + \lambda_{i-1}\left(1-e^{\lambda_{i+1}\frac{h}{2}}\right)\left(1-e^{\lambda_i\frac{h}{2}}\right)e^{\lambda_{i-1}\frac{h}{2}}$$
$$-\lambda_{i+1}\left(1-e^{\lambda_{i-1}\frac{h}{2}}\right)\left(1-e^{\lambda_i\frac{h}{2}}\right)e^{\lambda_i\frac{h}{2}} + \lambda_{i-1}\lambda_i\frac{h}{2}\left(1-e^{\lambda_{i+1}\frac{h}{2}}\right)e^{(\lambda_{i-1}+2\lambda_i)\frac{h}{2}}$$
$$-\lambda_i\lambda_{i+1}\frac{h}{2}\left(1-e^{\lambda_{i-1}\frac{h}{2}}\right) + \lambda_{i-1}\lambda_{i+1}\frac{h}{2}\left(1-e^{\lambda_i\frac{h}{2}}\right)^2 e^{\lambda_{i-1}\frac{h}{2}}$$

$$b_p = -\lambda_i\left(1-e^{\lambda_{i-1}\frac{h}{2}}\right)\left(1-e^{\lambda_{i+1}\frac{h}{2}}\right)e^{\lambda_i h} - \lambda_{i-1}\left(1-e^{\lambda_{i+1}\frac{h}{2}}\right)\left(1-e^{\lambda_i\frac{h}{2}}\right)e^{\lambda_{i-1}\frac{h}{2}}$$
$$-\lambda_i\lambda_{i+1}\frac{h}{2}\left(1-e^{\lambda_{i-1}\frac{h}{2}}\right) + \lambda_{i-1}\lambda_{i+1}\frac{h}{2}\left(1-e^{\lambda_i\frac{h}{2}}\right)e^{\lambda_{i-1}\frac{h}{2}}$$

Similar expressions for the two boundaries can be obtained in the same manner for the boundaries specified in definition 2.21. The boundary points have to be treated as special cases since these represent the breaking of the translational invariance symmetry of the differential equations.

Equation (5.1.6) is the discrete expression generated from applying the linear exponential spline interpolation locally. When the index i varies from 3 to $n-2$, equation (5.1.6) with appropriate boundary conditions creates a discretized solving system which satisfies exactly the same requirements as in the linear exponential spline scheme discussed in Chapter 3. A close examination of this system determines the relations between the node point velocity values and the residuals. The linearized time derivative terms are identical to those used in the program. Thus, this discretized solving system with equation (5.1.6) for the interior is defined as the *equivalent solving process* for the linear exponential spline method discussed in Chapter 3 so that the accuracy of the linear exponential spline scheme can be obtained by studying this equivalent solving process. Stability is studied in section 5.2 using the von Neumann stability theory.

Now, the accuracy of the linear exponential spline interpolation can be approached from studying equation (5.1.6) for the interior points. The algebra was performed using standard symbolic manipulation software. For the equation (5.1.6), the error estimation

can start by expressing u_{i-1}, u_{i+1} and all the exponential terms in Taylor expansion up to the third power of h:

$$u_{i-1} = u_i - u_i'h + \frac{u_i''}{2!}h^2 - \frac{u_i'''}{3!}h^3 \tag{5.1.7}$$

$$u_{i+1} = u_i + u_i'h + \frac{u_i''}{2!}h^2 + \frac{u_i'''}{3!}h^3 \tag{5.1.8}$$

$$e^{\lambda h} = 1 + \lambda h + \frac{(\lambda h)^2}{2!} + \frac{(\lambda h)^3}{3!} \tag{5.1.9}$$

The quantities r_{i-1}, r_{i+1} are also expanded in Taylor series with:

$$\begin{aligned} r_{i-1} &= r_i - r_{x_i}h + r_{xx_i}\frac{h^2}{2} - r_{xxx_i}\frac{h^3}{6} \\ r_{i+1} &= r_i + r_{x_i}h + r_{xx_i}\frac{h^2}{2} + r_{xxx_i}\frac{h^3}{6} \end{aligned} \tag{5.1.10}$$

Then the original differential equation: $r_i = u_i u_{x_i} - \varepsilon u_{xx_i}$ is recovered with an error term that is of second order in h:

$$u_i u_{x_i} - \varepsilon u_{xx_i} = r_i + O\left(h^2\right) \tag{5.1.11}$$

The same process can be applied on the boundaries to get the accuracy estimation for the two boundaries.

Thus the accuracy for the interior local spline interpolation is of second order in h. From the best approximation theorem for the generalized splines, it may be concluded that the linear exponential spline solving procedure has an accuracy which is no lower than second order, but with a lower absolute error. The computations confirm this as seen in Figure 5.1.11.

5.1.2 Accuracy study for the quadratic exponential spline scheme

There are also three steps involved when the quadratic exponential spline method is applied in solving Burgers equation. These were outlined in Chapter 3 as:

(i) calculating the second derivatives from equation (3.2.11);

(ii) calculating the first derivatives from equation (3.2.12);

(iii) solving the differential equation by time marching with the available derivatives from step (i) and (ii).

The accuracy study is again carried out accordingly. First, the accuracy of the first and second derivatives are obtained by Taylor expansion of related terms in equations (3.2.11) and (3.2.12) and then the resulting derivative terms are substituted back into the differential equation to get the overall spatial accuracy for the scheme.

Equation (3.2.11) is used to investigate the accuracy of the first derivative term. It is quoted below as equation (5.1.12):

$$As'_{i-1} + Bs'_i + Cs'_{i+1} = D(u_i - u_{i-1}) + E(u_i - u_{i-1}) \qquad (5.1.12)$$

where:

$$A = \frac{\left[2\left(1-e^{-\lambda_i h}\right)+\left(-2+\lambda_i h\right)\lambda_i h\right]}{h\left[-2\left(1-e^{-\lambda_i h}\right)+\lambda_i h\left(1+e^{-\lambda_i h}\right)\right]}$$

$$B = \frac{\left[-2\left(1-e^{-\lambda_i h}\right)+\left(2e^{-\lambda_i h}+\lambda_i h\right)\lambda_i h\right]}{h\left[-2\left(1-e^{-\lambda_i h}\right)+\lambda_i h\left(1+e^{-\lambda_i h}\right)\right]} - \frac{\left[-2\left(1-e^{\lambda_{i+1} h}\right)+\left(-2e^{\lambda_{i+1} h}+\lambda_{i+1} h\right)\lambda_{i+1} h\right]}{h\left[2\left(1-e^{\lambda_{i+1} h}\right)+\lambda_{i+1} h\left(1+e^{\lambda_{i+1} h}\right)\right]}$$

$$C = \frac{\left[2\left(1-e^{\lambda_{i+1} h}\right)+\left(2e^{\lambda_{i+1} h}+\lambda_{i+1} h\right)\lambda_{i+1} h\right]}{h\left[2\left(1-e^{\lambda_{i+1} h}\right)+\lambda_{i+1} h\left(1+e^{\lambda_{i+1} h}\right)\right]}$$

$$D = \frac{2\lambda_i\left(-1+e^{-\lambda_i h}+\lambda_i h\right)}{h\left[-2\left(1-e^{-\lambda_i h}\right)+\lambda_i h\left(1+e^{-\lambda_i h}\right)\right]}$$

$$E = \frac{2\lambda_{i+1}\left(1-e^{\lambda_{i+1} h}+\lambda_{i+1} h\right)}{h\left[2\left(1-e^{\lambda_{i+1} h}\right)+\lambda_{i+1} h\left(1+e^{\lambda_{i+1} h}\right)\right]}$$

In order to obtain an expression without too much complexity, some algebraic simplifications are applied to (5.1.18) by factoring out the denominators. First, $e^{\lambda_i h}$ is multiplied to the numerator and the denominator of the coefficients A and D and to those of the first part of the coefficient B. Then, the whole equation is multiplied by $h\left[-2\left(1-e^{-\lambda_i h}\right)+\lambda_i h\left(1+e^{-\lambda_i h}\right)\right]\left[2\left(1-e^{\lambda_{i+1} h}\right)+\lambda_{i+1} h\left(1+e^{\lambda_{i+1} h}\right)\right]$. Then, s'_{i-1} and s'_{i+1} on the left hand side, u_{i-1} and u_{i+1} on the right hand side and all the exponential terms on both sides are expanded into Taylor series up to the third power of h while assuming h small. After truncating the high order terms, an expression for s'_i is obtained as:

$$s_i' = u_i' + \frac{(\lambda_p - \lambda_i)u_i'''}{144}h^3 + O(h^4) \tag{5.1.13}$$

This equation shows that in the matrix set up from equation (3.2.11), the first derivative is third order accurate with respect to h. If the same procedure is applied to equation (3.2.12), equation (5.1.14) is recovered:

$$as_{i-1}'' + bs_i'' + cs_{i+1}'' = \frac{1}{h}(u_{i-1} - 2u_i + u_{i+1}) \tag{5.1.14}$$

$$a = \frac{\left[2\left(1 - e^{-\lambda_i h}\right) + (-2 + \lambda_i h)\lambda_i h\right]}{2\left(-1 + e^{-\lambda_i h}\right)\lambda_i^2 h}$$

$$b = \frac{\left[2\left(1 - e^{-\lambda_i h}\right) + \left(-2 + \lambda_i h e^{-\lambda_i h}\right)\lambda_i h\right]}{2\left(-1 + e^{-\lambda_i h}\right)\lambda_i^2 h} + \frac{\left[2\left(1 - e^{\lambda_{i+1} h}\right) + \left(2 + \lambda_{i+1} h e^{\lambda_{i+1} h}\right)\lambda_{i+1} h\right]}{2\left(-1 + e^{\lambda_{i+1} h}\right)\lambda_{i+1}^2 h}$$

$$c = \frac{\left[2\left(1 - e^{\lambda_{i+1} h}\right) + (2 + \lambda_{i+1} h)\lambda_{i+1} h\right]}{2\left(-1 + e^{\lambda_{i+1} h}\right)\lambda_{i+1}^2 h}$$

The accuracy for the second derivative is obtained in the form:

$$s_i'' = u_i'' + \frac{(\lambda_i + \lambda_p)u_i'''}{24}h^2 + O(h^3) \tag{5.1.15}$$

Thus, the second derivative is second order accurate with respect to h. It is also noticed that the leading error terms in both equations (5.1.13) and (5.1.15) are functions of the third derivatives.

The time marching step uses a backward Euler method which is of first order in the time step size, Δt:

$$u_i^{n+1} = u_i^n + r\Delta t + O(\Delta t^2)$$
$$r = \left(\frac{\partial u}{\partial t}\right)^n \tag{5.1.16}$$

where the subscript n indicates the time level. The time step size, Δt, is related to the spatial grid size, h, via the CFL number [1].

If equations (5.1.13), (5.1.15) and (5.1.16) are substituted back into the differential equation, the overall accuracy for this quadratic exponential spline method is obtained in the form:

$$u_t + uu_x - \varepsilon u_{xx} = O(\Delta t) + O\left(h^2\right) \tag{5.1.17}$$

It is of second order accuracy with respect to the spatial grid size and first order accuracy with respect to the time step size. This result, along with the corresponding linear exponential spline result, suggests that it is the homogenous part of the Roscoe linearized equation that fixes the order of the method while the non-homogenous component controls the absolute error.

5.2 Stability study

The numerical results from the two exponential spline methods show unconditional stability with respect to the cell Reynolds number. The computational results shown in Figures 5.1.1 to 5.1.11 show that the two methods under study can work with arbitrary cell Reynolds number Udx/ε with reasonably accurate results while the second order central difference and fourth order compact scheme fails if $Re_{cell} > 2.5$. The second order cubic spline method does not converge when $Re_{cell} > 5$. The latter method loses monotonicity, as shown in Figure 5.1.4, when $Re_{cell} = 5$, thus violating the maximum property of Burgers equation discussed in Chapter 3. The original Roscoe scheme is also unconditionally stable with respect to cell Reynolds number, however, Figure 5.1.11 shows that it is not as accurate as the linear exponential spline method.

The stability of the linear exponential spline method is given below through a numerical von Neumann analysis [21] of the equivalent system defined in equation (5.1.6).

When a one-dimensional, two-level, three-point scheme is given by:

$$b_3 u_{i+1}^{n+1} + b_2 u_i^{n+1} + b_1 u_{i-1}^{n+1} = a_3 u_{i+1}^n + a_2 u_i^n + a_1 u_{i-1}^n \tag{5.2.1}$$

the corresponding amplification factor, G, is:

$$|G|^2 = \frac{A_1\sigma^2 + A_2\sigma + 1}{B_1\sigma^2 + B_2\sigma + 1} \tag{5.2.2}$$

where:

$$\sigma = \frac{\sin^2\phi}{2} \qquad A_1 = 16a_1a_3 \qquad B_1 = 16b_1b_3$$

$$A_2 = 4\left[(a_3 - a_1)^2 - (a_3 + a_1)\right] \qquad B_2 = 4\left[(b_3 - b_1)^2 - (b_3 + b_1)\right]$$

For stability it is necessary to have:

$$|G| \le 1 \tag{5.2.3}$$

A numerical study of the equivalent solving process of the linear exponential spline scheme shows that, for arbitrary cell Reynolds numbers, the condition (5.2.3) is satisfied so that the linear exponential spline scheme is unconditionally stable with respect to cell Reynolds number.

The interior point relation for the equivalent solving process is defined in equation (5.1.6). For Burgers equation, the residual r is defined as: $r_j = -\left(\frac{\partial u}{\partial t}\right)_j$. When the backward Euler time marching is applied, this becomes:

$$r_j = -\frac{u_j^{n+1} - u_j^n}{\Delta t} \tag{5.2.4}$$

Substitution of equation (5.2.4) into equation (5.1.6) and rearranging the terms gives:

$$\begin{aligned} &-\frac{b_m}{\Delta t U_m}u_{i-1}^{n+1} - \frac{b_i}{\Delta t U_i}u_i^{n+1} - \frac{b_p}{\Delta t U_p}u_{i+1}^{n+1} \\ &= \left(a_m - \frac{b_m}{\Delta t U_m}\right)u_{i-1}^n + \left(a_i - \frac{b_i}{\Delta t U_i}\right)u_i^n + \left(a_p - \frac{b_p}{\Delta t U_p}\right)u_{i+1}^n \end{aligned} \tag{5.2.5}$$

The time step size can be eliminated form this expression by relating it to the Courant number, here denoted as CFL. Recall that the time step size is related to the CFL time step size by $\Delta t = CFL * \Delta t_{CFL}$ where $\Delta t_{CFL} = (h/U)_{min}$. The minimum will occur where U has its maximum magnitude, or, U_{max}. Then $\Delta t = CFL * \frac{h}{U_{max}}$ and equation (5.2.5) can be written as:

$$\begin{aligned} &-\frac{U_{max}}{h\,CFL}\left[\frac{b_m}{U_m h}u_{i-1}^{n+1} + \frac{b_i}{U_i h}u_i^{n+1} + \frac{b_p}{U_p h}u_{i+1}^{n+1}\right] \\ &= \left(a_m - \frac{U_{max}}{h\,CFL}\frac{b_m}{U_m h}\right)u_{i-1}^n + \left(a_i - \frac{U_{max}}{h\,CFL}\frac{b_i}{U_i h}\right)u_i^n + \left(a_p - \frac{U_{max}}{h\,CFL}\frac{b_p}{U_p h}\right)u_{i+1}^n \end{aligned} \tag{5.2.6}$$

The ratios of the velocities may be replaced with the ratios of the λ's since the ε's cancel. Thus the equation may be put into the form:

$$-\frac{\lambda_{max}}{h\,CFL}\left[\frac{b_m}{\lambda_m}u_{i-1}^{n+1}+\frac{b_i}{\lambda_i}u_i^{n+1}+\frac{b_p}{\lambda_p}u_{i+1}^{n+1}\right]$$
$$=\left(a_m-\frac{\lambda_{max}}{h\,CFL}\frac{b_m}{\lambda_m}\right)u_{i-1}^{n}+\left(a_i-\frac{\lambda_{max}}{h\,CFL}\frac{b_i}{\lambda_i}\right)u_i^{n}+\left(a_p-\frac{\lambda_{max}}{h\,CFL}\frac{b_p}{\lambda_p}\right)u_{i+1}^{n} \qquad (5.2.7)$$

where λ_{max} is the value of λ when U =U_{max}.

The coefficients in equation (5.2.1) then become:

$$b_1=-\frac{\lambda_{max}b_m}{\lambda_m h\,CFL} \qquad a_1=\left(a_m-\frac{\lambda_{max}b_m}{\lambda_m h\,CFL}\right)$$
$$b_2=-\frac{\lambda_{max}b_i}{\lambda_i h\,CFL} \qquad a_2=\left(a_i-\frac{\lambda_{max}b_i}{\lambda_i h\,CFL}\right) \qquad (5.2.8)$$
$$b_3=-\frac{\lambda_{max}b_p}{\lambda_p h\,CFL} \qquad a_3=\left(a_p-\frac{\lambda_{max}b_p}{\lambda_p h\,CFL}\right)$$

where a_m, a_i, a_p, b_m, b_i and b_p are defined as in equation (5.1.6). By multiplying the entire equation by h, every λ can be rewritten as some cell Reynolds number. To evaluate λ_{max}, a multiplying factor ($\geq$1) on max[λ_m, λ_i, λ_p] can be used. If this factor is equal to 1, then the local values of λ, and hence the local values of the velocity, are near the maximum value corresponding to a convection dominated region. If the multiplying factor is large, this corresponds to the local conditions being more diffusion dominated. Since convection regions are less stable numerically, the more critical values of λ_{max} will be those for which the multiplying factor is near 1. In each of the cases to be shown, a value of the multiplier of 1 was used. When the factor was increased, each curve on the polar graph tended toward the circle. It can be shown, that in the limit as the multiplier gets large, this corresponds to the a's going to zero and all stability curves becoming coincident with the unit circle. These cases, then become neutrally stable at the limit, but have very small error.

With the coefficients defined by equation (5.2.8), the amplification factor, G, can be evaluated. If the norm of G is less than or equal to one for all phase angles, ϕ, then the numerical scheme is stable. The typical presentation of a stability study is by a polar representation of G versus phase angle (the axes are labeled as Re and Im for Real and Imaginary since the von Neumann analysis finds G as a complex quantity). The

resulting curve is compared with the unit circle. The G curve remaining inside or on the circle then represents stability. The quantities are calculated numerically for a range of cell Reynolds numbers and Courant numbers. The results are shown in the phase planes in Figures (5.2.1) and (5.2.2).

Figure (5.2.1) shows the stability for a large range of cell Reynolds numbers (-50 to 50) for CFL numbers of .1, .5, and 1. These CFL numbers were chosen due to the explicit marching procedures being used for the Burger equation studies which is subject to the well-known time step limitation of CFL ≤ 1. In each case it is seen that the numerically-computed values of the norm |G| are less than or equal to unity. The case of cell Reynolds number equal to zero results in the magnitude of G of unity for all phase angles.

For the range of cell Reynolds numbers in Figure (5.2.1), stability with respect to cell Reynolds numbers is retained for any CFL ≤ 1. However, with cell Reynolds numbers near one, the CFL number had to be reduced for stability. This is shown in Figure (5.2.2) where a range of cell Reynolds numbers near one are used. For CFL = .25, a cell Reynolds number of .5 gives a norm of the amplification factor greater than one as shown. For CFL = .1, all magnitudes are less than or equal to one. With experimentation it was found that, for all values of the cell Reynolds number tested, a CFL could always be found where all magnitudes of G were less than or equal to one. In this case, the first part of Figure 5.2.3, CFL = .1 shows complete stability.

The results in Figures (5.2.1) and (5.2.2) were for constant values of the cell Reynolds number at all points. One of the unique features of the Roscoe type schemes is that they locally adjust to changing cell Reynolds numbers. To test the stability in such a circumstance, a second study was performed where the cell Reynolds numbers were varied at the three points. This is shown in Figure (5.2.3). A linear variation in cell Reynolds number was used for each case. The center value was varied from -9 to 9 and the other two values were modified by factors of .5 and 1.5. The actual values of the three cell Reynolds numbers is given in the legend of the figure. The first number is that for the cell Reynolds number at the (i-1)st grid point, the second is at i, and the third is at i+1. The CFL number for complete stability was .25.

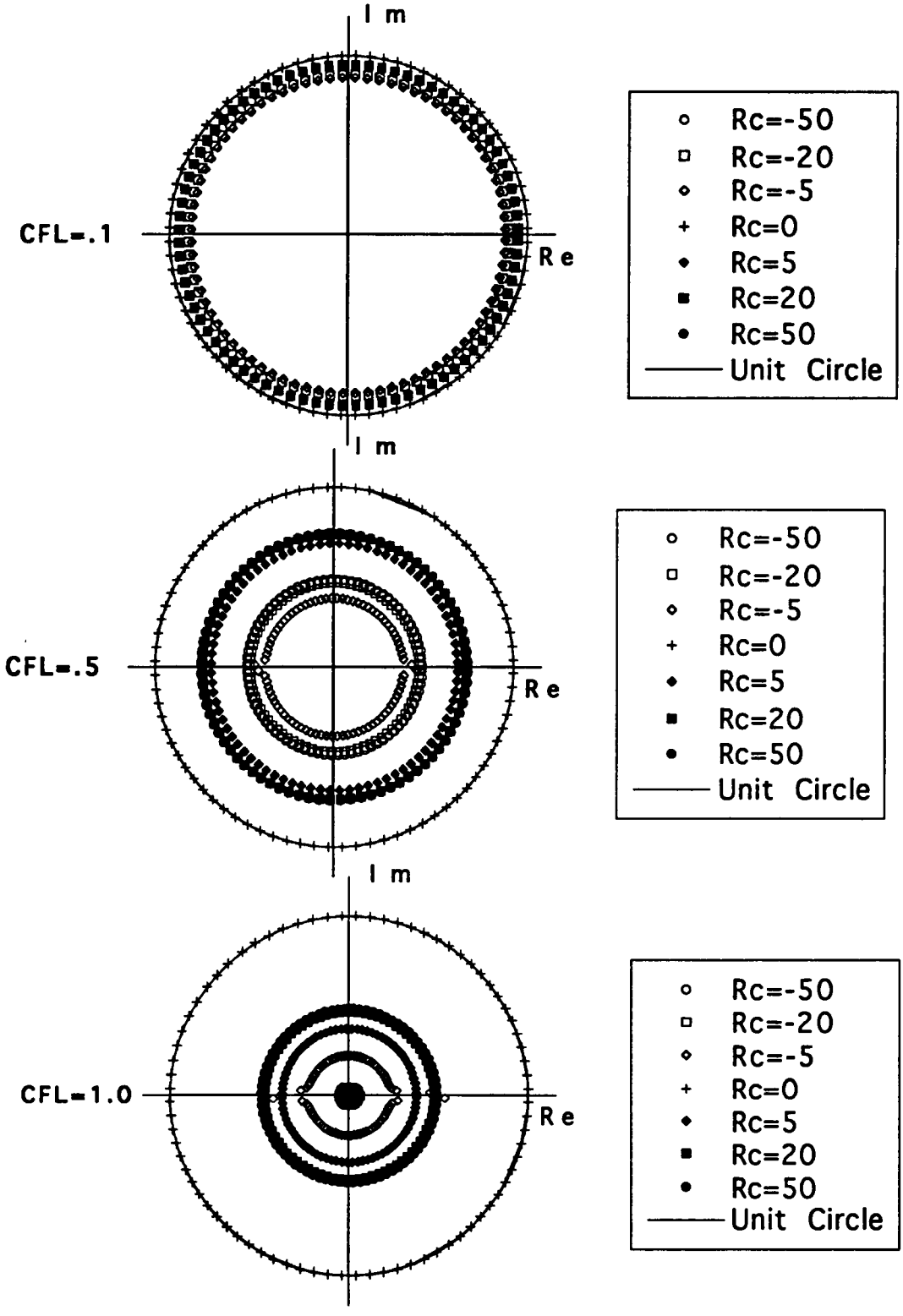


Figure 5.2.1 Polar Plot of |G| for the Linear Exponential Spline Procedure at Large Range of Cell Reynolds Number

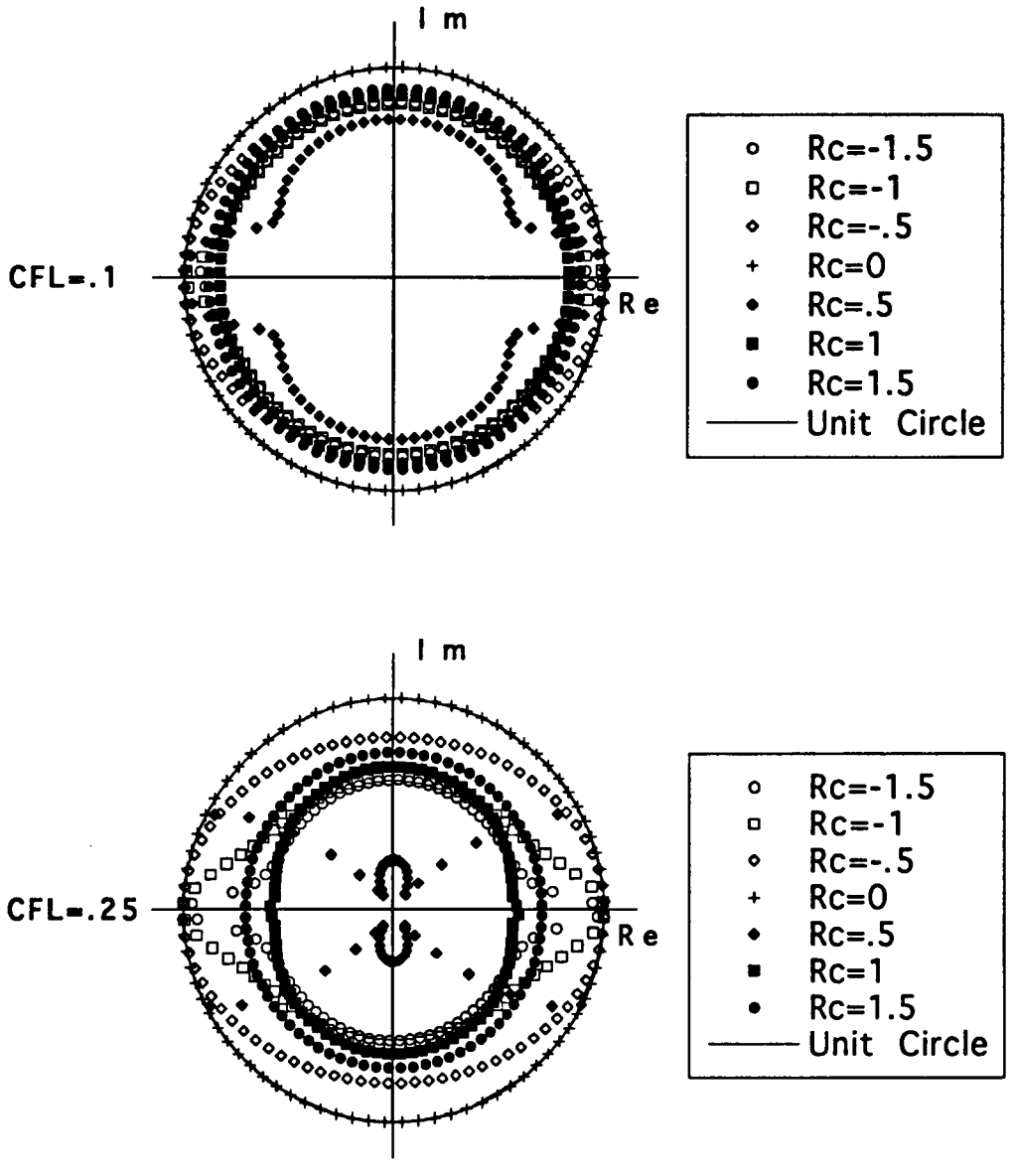


Figure 5.2.2 Polar Plot of |G| for the Linear Exponential Spline Procedure at Small Range of Cell Reynolds Number

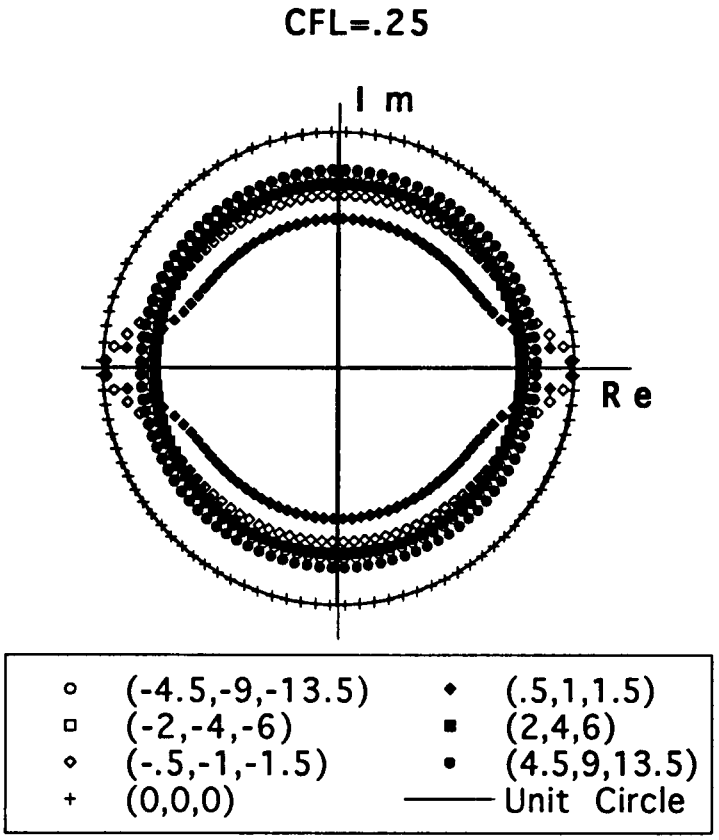


Figure 5.2.3 Polar Plot of |G| for the Linear Exponential Spline Procedure for variable cell Reynolds number.

The last series of numerical stability studies was carried out for the case of a sign change in the local values of the cell Reynolds number. In this case the center value of the cell Reynolds number was set to zero and the other two were additive inverses (same value, different sign). This simulates the shock-like behavior of Burgers equation for the case studied here. The result was that the norm of the amplification factor remained directly on the unit circle for all values of the CFL number. The figure is not given since it is merely the unit circle with the data directly on top.

The conclusion is that, for virtually all combinations of local cell Reynolds number, a CFL can always be found such that the norm of the amplification factor stays either within or on the unit circle. While this exercise does not constitute a rigorous analytical proof of unconditional stability, it can be inferred that, for the range of cell Reynolds numbers tested, the linear exponential spline methodology does provide a stable procedure.

5.3 Analysis of the Numerical Solutions

The numerical solutions obtained from the two exponential spline methods were shown in Figures 5.1.1 to 5.1.15 along with some computed results from the original Roscoe, the central difference, the fourth order compact and the cubic spline methods. The comparisons in these figures show several advantages and improvements that have been achieved by the generalized Roscoe procedure which generated the two exponential spline methods. These improvements include improved stability over traditional methods of finite differencing and improved accuracy both through the manner in which the generalized interpolating functions were generated and in the use of splines to create the finite difference equations.

5.3.1 Spatial Stability

Stability with respect to cell Reynolds number of two exponential spline solving processes were shown numerically in Figures 5.1.5 and 5.1.6 where $Re_{cell} = U\Delta x/\varepsilon$ was 50.0 and 500.0 respectively. The numerical von Neumann stability study also showed this result in Figures 5.2.1 to 5.2.3 for the linear exponential spline method. This capability of working with arbitrary cell Reynolds number is achieved without adding any so-called artificial viscosity terms. It is, then, a significant property of the generalized Roscoe discretization. Figures 5.1.5 and 5.1.6 showed that the original Roscoe method is also unconditionally stable, though it is less accurate than the present generalization. This spatial stability distinguishes these Roscoe type methods from the others.

5.3.2 Improvements in Accuracy

As mentioned there are two aspects of the method that should improve the accuracy over the original Roscoe method in particular and over traditional methods of the same order in general. These were the coupled manner in which the interpolating functions were generated and the use of splines to create the finite difference expressions. Since the computations did show improved accuracy over the other methods, it remains to assess the relative importance of both aspects.

The analysis presented in this work does not really demonstrate the improvements due to the system approach to integration. Rather, it has been intuitively argued that preserving as much of the nonlinear system properties as possible ought to allow a superior resolution of the nonlinear solution by the numerical method. Clearly, the original Roscoe method showed improved accuracy over central differencing by a factor

of around 3 for cell Reynolds numbers where central differencing was stable. Thus there is merit in even the basic coupling of convection-diffusion pairs. Since the present effort couples the entire system (although only one direction at a time), it would appear that this would present an even more accurate procedure. To show this will require a more careful analysis and is beyond the scope of this study. Thus, a true assessment of the accuracy improvement due to the system coupling alone remains to be accomplished.

Accuracy improvement due to the use of splines has a firmer mathematical basis. The best approximation theorem assumes that a spline interpolation using a certain interpolation function gives smaller absolute error than the non-spline interpolated solutions. Without the spline interpolation, the linear exponential spline method studied here is similar to the original Roscoe scheme (it is identical for the nonlinear Burgers equation). In Figure 5.1.11, the difference between the results from the linear exponential spline and the original Roscoe method shows very clearly the improvement gained by the spline interpolation. Although they both are of second order, the absolute error for the linear exponential spline results is much smaller in the norm used. Thus, it may be concluded that the linear exponential spline interpolation improves the absolute accuracy rather than raising the order of the accuracy. The spline interpolation imposes continuity conditions on the velocity and velocity derivatives (by matching the value and derivatives at the spline matching points) through the whole domain which helps to reduce the computational errors.

It is known from the previous chapters that the differences between the linear and quadratic exponential splines are:

(i) In the quadratic exponential spline interpolation method, the time derivative term is approximated to a higher order with respect to the spatial variable x.

(ii) In the spline interpolation, the quadratic exponential spline is required to match up to the second derivatives while the linear exponential spline can only match to the first derivative.

Theoretically, improved accuracy is expected from the quadratic exponential spline method since it has a higher order approximation of r, the forcing function in the linearized differential equation. However, Figure 5.1.11 shows that for the steady Burgers equation, the quadratic exponential spline is less accurate than the linear exponential spline method. In Figure 5.1.15, the quadratic exponential spline method appeared to have a lower order of accuracy than was shown in section 5.1.2. There are several possible explanations. One is the two extra constraints used to close the problem in the quadratic exponential spline interpolation (in this case, the zero second derivatives

at the two boundaries). Because there is no physical foundation for doing this, some error could be generated and the accuracy reduced.

Even though the quadratic exponential spline method still has some unsolved problems, Figure 5.1.15 shows that the quadratic exponential spline procedure with a higher order approximation of the Roscoe vector, r, displays the most accurate solution for the Lamb vortex problem. The reason why this does not show in solving the Burgers equation may be a result of the Burgers equation work being for a steady state problem. As the time derivative term, $r = -\partial u/\partial t$ becomes vanishingly small when the solution converges so that the higher order approximation of this r does not affect the final solution very much. In solving the unsteady problems like the unsteady Lamb vortex, on the other hand, the accuracy of r influences the results at every time level because it does not go to zero as time evolves.

The linear exponential spline method did not appear to work as well for the Lamb vortex problem as it did in solving the Burgers equation (see Figure 5.1.15). One of the reasons could be that in solving the pressure Poisson equation, this linear exponential spline made use of the Calhoon average technique to calculate the derivatives while all the other two methods use direct differentiation from the local spline interpolation functions.

5.3.4 The problems and recommendations for further study

The above studies have concluded that the Roscoe local linearization determines the order of the accuracy while the spline interpolation improves absolute error values. This may suggest that a higher order approximation be applied in the linearization of the nonlinear coefficients in the differential equations. For example, if Burgers equation (2.2.1) is considered, the Taylor expansions of the nonlinear coefficients is:

$$a = u(x,t) = u_i + \left(\frac{\partial u}{\partial x}\right)_i (x - x_i) + \left(\frac{\partial u}{\partial t}\right)_i^n \left(t - t^n\right) + \ldots \qquad (5.3.1)$$

Instead approximating 'a' by u_i, the constant local velocity value, the first two terms in equation (5.3.1) can be used to represent the coefficient 'a', that is:

$$a = u(x,t) = u_i + \left(\frac{\partial u}{\partial x}\right)_i (x - x_i) \qquad (5.3.2)$$

Then, substitution of equation (5.3.2) back to the original nonlinear differential equation so that the local linearized equation has a different formula than the one studied in the present research. Then the local analytical solutions to this new linearized system can be used as interpolation functions to create the discrete model for the solving procedure. It can be expected that the corresponding phase space for this new linearized system will be different from the one studied in section 2.3. Hopefully, the order of the slope and curvature of this new phase space trajectory can be raised. Then the diffusion error in expression (2.3.22) will decreased so that a higher order of the accuracy can be expected.

When the forcing function r in equation (2.2.2) was approximated to $\alpha+\beta x$, there were no constraints on the two unknown constants α and β. Theoretically, the time derivative term, $r=-\frac{\partial u}{\partial t}(x,t)$, should be continuous with respect to spatial independent variable x at each time level. Moreover, in solving multidimensional problems, some procedure should be devised which allows coupling between the differential operators. These considerations could lead to reasonable requirements on these undetermined constants.

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

A new discretization method for solving systems of differential equations, the generalized Roscoe numerical procedure, has been discussed in detail. During the development of this method, the emphasis has been on trying to build into the interpolation functions as much of the character of the system of equations as possible. The method includes:

(1) linearization of the original nonlinear differential equation system on each grid interval;

(2) a set of unique analytical solutions are obtained from step (1), one for each dependent variable;

(3) using these analytic functions as a spline basis to evaluate the integration constants;

(4) directly differentiating the spline function found in step (3) to obtain the finite difference representations of the original nonlinear system.

The generalized Roscoe method attempts to retain the properties of the given system. In steps (1) and (2), the local information from the original nonlinear system is imposed on the node points by requiring them to satisfy the analytical solution to the locally linearized equation. Step (3) evaluates the constants of integration by satisfying spline constraint relations. This not only imposes the constraint that the analytic function pass through node values, but also that the piecewise continuous set of functions be continuous to some degree at grid cell edges. In this way, the local linear information on each grid interval is collected together to present the nonlinearity for the original problem in a discrete fashion. In step (4), the completely determined intepolating functions can be directly differentiated locally to find expressions for such derivatives of the dependent variables as are found in the original nonlinear equations. The phase space of one of the test problems shows that this method approximates the nonlinear system in a discrete fashion.

It was shown both analytically and numerically that the present version of the generalized Roscoe procedure gives second order accuracy with respect to the spatial grid size.

So far, the two numerical schemes were developed from the generalized Roscoe method, the linear and quadratic exponential spline schemes. At present, the preliminary

results indicate that one or the other may be optimal in solving different problems. In steady state problems, the linear exponential method is probably to be recommended since it gave the best accuracy compared to other second order methods in the test case. If the problem under study is unsteady, the quadratic method appears to be more favorable since it demonstrated superior accuracy of the other methods, particularly on coarser grids.

There are many unanswered questions with the two exponential schemes. For example, why does the quadratic exponential spline method not have better accuracy than the linear one? Can this method be raised to a higher order by applying higher order approximation in the local linearization?

Some recommendations for future study include the need to obtain a closer link between directions in a multidimensional problem and the need to investigate the coupling between the spatial and temporal discretizations. It is felt that more information is yet to be obtained from phase space studies. In particular, the stability properties of an algorithm should be reflected in phase space. Higher order approximations of the nonlinear coefficients in the given equations is also recommended for further study.

The final point to be made is that the generalized Roscoe procedure is applicable to virtually any system of differential equations. Because the procedure is able to build into the finite difference expressions many properties from the original system which were not routinely retained previously, it represents an important advance in numerical approximation.

LIST OF REFERENCES

LIST OF REFERENCE

[1] Timan, A. F., Theory of Approximation of Functions of a Real Variable, Pergamon Press Ltd., Oxford, England, 1963 (translated from the original 1960 Russian edition by J. Berry).

[2] Anderson, Dale A., Tannehill, J. C. and Pletcher, R. H., Computational Fluid Mechanics And Heat Transfer, Hemisphere Publishing Co., New York, 1984.

[3] El-Mistikawy, T. M. and Werle, M. J., " Numerical Methods For Inviscid/Viscous Flows", Report No. AFL 77-9-34, Dept. of Aerospace Engineering and Applied Mechanics, University of Cincinnati, Sept., 1977.

[4] Roscoe, D. F, " New Methods For The Derivation Of Stable Difference Representations For Differential Equations", J. Inst. Maths Applics (1975) 16, pp 291-301.

[5] Roscoe, D. F., " The Application of Matrix Theorems To The Construction Of Stable Difference Representations Of The Equations Of Fluid Dynamics", Lin. Alg. Applics., 14 (1976) pp 123-130.

[6] Roscoe, D. F., " The Numerical Solution Of The Three- Dimensional Viscous Navier- Stokes Equations", Ph.D. Thesis, University of Aston, Birmingham, in Great Britain, 1974.

[7] Allen, D.N. de G. and Southwell, R. V., "Relaxation Method Applied To Determine The Motion, In Two Dimensions, Of A Viscous Fluid Past A Fixed Cylinder", Quarterly J. of Mechanics and Applied Mathematics, Vol. VIII, Part 2, 1955, pp.129-145.

[8] Spalding, D. B., " A Novel Finite Difference Formulation For Differential Expressions Involving Both First And Second Derivatives", Inter. J. for Numer. Methods in Engrg., V. 4, 1972, pp. 551-559.

[9] Chien, John C., " A General Finite-Difference Formulation with Application to Navier-Stokes Equations", Computers and Fluids. Vol. 5,1977, pp.15-31.

[10] Roach, R.L., " A Consistent Finite-Difference Derivation For The Navier-Stokes Equations", AIAA-82-04102, AIAA 20th Aerospace Sciences Meeting, Orlando, FL, Jan 1982.

[11] Roach, R.L. and Cahoon, W.H., "A Uniform Differencing Procedure Applied To the Incompressible Navier-Stokes equations", AIAA-91-0238, AIAA 29th Aerospace Sciences Meeting, Reno NV 1991.

[12] Calhoon, W. H., Jr. and Roach, R. L., "A New Upwind Procedure For The Incompressible Navier-Stokes Equations On Non-Stagged Grides", AIAA-93-0538, AIAA 31st Aerospace Sciences Meeting, Reno NV, Jan. 1993.

[13] Campbell, S. L. and Meyer, C. D., Generalized Inverses Of Linear Transformations, Pitman Publishing, Ltd., 1979 [reissued by Dover 1991].

[14] Prenter, P. M., Splines And Variational Methods, John Wiley & Sons, Inc. 1975

[15] Ahlberg, H. H., E. N. Nilson, and Walsh, J. L., The Theory Of Splines And Their Applications, Academic Press, New York, 1967.

[16] Rubin, S. G. and Graves, R. A., "Viscous Flow Solutions With A Cubic Spline Approximation", Computers and Fluids, Vol. 3, 1975, pp. 1-36.

[17] Kreiss, H-O and Lorenz, J, Initial Boundary Value Problems and the Navier Stokes Equations, Academic Press, 1989.

[18] Wu, J. Z., Ma, H. Y. and Zhou, M. D., Introduction to Vorticity and Vortex Dynamics, Advanced Education Press, Beijing, 1993.

[19] Abdallah, S., "Numerical Solutions For The Incompressible Navier-stokes Equations In Primitive Variables Using Non-Staggered grid, II", Journal of Computational Physics, Vol. 70, pp. 193-202, 1987.

[20] Horn, Roger A. and Johnson, Charles, Matrix Analysis, Cambridge University Press, New York, 1987.

[21] Hirsch, C., Numerical Computation of Internal and External Flows, volume 1: Fundamentals of Numerical Discretization, John Wiley & Sons Ltd., Chichester, 1988.

VITA

Yanping Shi came from Beijing, China. Her parents still reside in Beijing where her father, Shulin Shi, is a retired industrial manager, and her mother, Guiying Sun, is a retired pharmacist. Yanping grew up in Beijing with a brother and a sister. After graduation from middle School, she attended Beijing Institute of Technology, in Beijing, China, where she received her Bachelor of Science degree in Aerospace Engineering in July, 1983. Following her graduation, she worked in Qinhua Institute of Mechanics and Electronics, in Xian, China. After three year working in a area of fight vehicle system design, she went to back to Beijing Institute of Technology to continue her graduate study. She finished her graduate study and received a Master of Science degree in Aerospace Engineering in May, 1989. Then she became a system engineer in the Research and Development department in Ministry of Mechanics and Electronics, Beijing, China. In June, 1993, Yanping entered graduate study in Aerospace Engineering at the University of Tennessee Space Institute, in Tullahoma, Tennessee. Here, she worked as a Graduate Research Assistant to Dr. Robert Roach. The primary area of her research is the computational fluid dynamics. Yanping received her Master of Science degree in Aerospace Engineering from the University of Tennessee, in Knoxville, Tennessee, in May 1995.