AN INVESTIGATION OF THE
UPWIND LEAPFROG METHOD FOR
SCALAR ADVECTON AND
ACOUSTIC/AEROACOUSTIC WAVE
PROPAGATION PROBLEMS

by

Jeffrey Paul Thomas

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering and Scientific Computing)
in The University of Michigan
1996

Doctoral Committee:
Professor Phillip L. Roe, Chairperson
Associate Professor Robert Krasny
Associate Professor Kenneth G. Powell
Professor Bram van Leer
To mom and dad for years of encouragement and support
ACKNOWLEDGEMENTS

I would first like to thank the Department of Education for providing a PhD fellowship that supported me during the first three years of my graduate study at the University of Michigan. Next, I wish to thank the Department of Aerospace Engineering at the University of Michigan for providing graduate teaching assistantship positions for the following two years that enabled me to complete my PhD program. I would also like to Thank NASA for financial support under contract number NASA-G-NAG-1-1514 that enabled me to keep working during the summer months for a number of years in my program. Finally, I would like to throughly thank Professor Philip Roe for starting me off on the topic of my dissertation in addition to providing uncountable hours of support and encouragement involved in the research for this dissertation.
A number of years ago when started on this project, I really had no idea of what the traits of a “good” numerical scheme for aeroacoustics should comprise. To start off, like a number of other researchers in this newly developing field of research, I “toyed” around with some the “state-of-the-art” numerical methods designed for steady-state compressible flow problems. Although these schemes can solve the time-dependent equations, a number of mechanisms are incorporated to accurately capture steady-state shocks. These mechanisms (upwinding, limiting, artificial dissipation, etc.) tend to be dissipative in order to keep the scheme stable and to prevent oscillations around shocks. As I quickly found out for the simpler linearized acoustic equations where one would like to propagate waves over long distances, these schemes were much too dissipative. The amplitude of initial data tended to be drastically attenuated after traveling just a few wave lengths.

My thesis advisor professor Philip Roe, who was already aware of this problem, came to me with a relatively new and generally unknown scheme known as the upwind leapfrog method. The most attractive feature of this scheme is at least for the case of simple scalar advection, it is totally free of dissipation. Professor Roe suggested I look into the possibility of expanding this scheme to the more general aeroacoustic equations, and this is where the research that has gone into this dissertation basically started.
# TABLE OF CONTENTS

DEDICATION ......................................................... ii

ACKNOWLEDGEMENTS ............................................. iii

PREFACE .......................................................... iv

LIST OF FIGURES ................................................ vii

LIST OF TABLES .................................................... xi

LIST OF APPENDICES ............................................. xii

CHAPTER

I. INTRODUCTION .................................................. 1

1.1 Initiative ....................................................... 2

1.2 Governing and Model Equations ................................ 2

1.2.1 Aeroacoustic Equations .................................... 4

1.2.2 Acoustic Equations .......................................... 5

1.2.3 Scalar Advection Equation .................................. 7

1.2.4 Time or Frequency Domain .................................. 7

1.3 Numerical Methodology ........................................ 8

1.3.1 Numerical Error Considerations ............................ 12

1.3.2 Computational Cost Considerations ........................ 14

1.4 Objectives and Organization ................................... 17

II. METHOD DEVELOPMENT FOR SCALAR ADVECTION ............ 20

2.1 Design of the Upwind Leapfrog Scheme for Scalar Advection . 21

2.1.1 Interpretation as Interpolation ............................. 22

2.1.2 Equivalent Differential Equation Approach ................ 26

2.1.3 A Conservative Formulation ................................. 36

2.2 Error Analysis .................................................. 38

2.2.1 Dispersion Relations ......................................... 38

2.2.2 Group Velocity Considerations ............................. 42

2.2.3 Performance Comparisons ................................. 45
2.3 Higher-Order Extensions ........................................ 48
   2.3.1 Fourth-Order Upwind Leapfrogs .......................... 48
   2.3.2 Fourth-Order Performance Comparisons ................. 53

III. MORE GENERAL FORMS OF SCALAR ADVECTION ........ 55

3.1 Non-constant Advection Speed ............................... 55
3.2 Source Terms ................................................ 60
   3.2.1 Spherical Wave Equation ............................... 63
   3.2.2 Higher-Order Accuracy ................................. 64
3.3 Non-constant Coefficients and Source Terms .............. 66
3.4 Multiple Dimensions ......................................... 71
   3.4.1 Mesh and Stencil Strategies ........................... 71
   3.4.2 Dispersion Analysis .................................... 75
   3.4.3 Non-Constant Coefficients and Fourth-Order Accuracy 79
   3.4.4 Solid Body Rotation Problem .......................... 83
   3.4.5 Multiple Dimensions with a Source Term ............... 85

IV. ACOUSTICS IN STATIONARY AND NON-STATIONARY MEDIA .... 91

4.1 Method Development for Acoustics in a Stationary Medium . 91
   4.1.1 Bicharacteristic Approach ............................ 92
   4.1.2 Dispersion Analysis .................................... 98
   4.1.3 Higher-Resolution Methods ............................ 105
4.2 Method Development for Acoustics in a Non-stationary Medium 112
4.3 Upwind Leapfrog Bicharacteristic Approach .................. 114
4.4 Higher-Resolution Formulation .............................. 120
   4.4.1 Wall Reflection Problem ............................... 122

V. TOWARDS THE AEROCOUSTIC SYSTEM ..................... 129

5.1 One-Dimensional Formulations .............................. 129
5.2 Two-Dimensional Formulations .............................. 133

VI. CONCLUSIONS AND RECOMMENDATIONS .................... 139

APPENDICES .................................................. 145

BIBLIOGRAPHY ................................................ 153
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Numerical Error Types for Time-Dependent Hyperbolic Equations</td>
<td>13</td>
</tr>
<tr>
<td>2.1</td>
<td>Stencil Arrangements for Three Classical Second-Order Techniques along with Upwind Leapfrog Scheme for the Scalar Advection Equation</td>
<td>23</td>
</tr>
<tr>
<td>2.2</td>
<td>A Conservative Formulation</td>
<td>37</td>
</tr>
<tr>
<td>2.3</td>
<td>Phase Speed Error Trends for the Leapfrog Schemes to the Scalar Advection Equation</td>
<td>43</td>
</tr>
<tr>
<td>2.4</td>
<td>Second-Order Scheme Numerical Results to Model Scalar Advection Problem</td>
<td>46</td>
</tr>
<tr>
<td>2.5</td>
<td>Fourth-Order Upwind Leapfrog Techniques for Scalar Advection</td>
<td>49</td>
</tr>
<tr>
<td>2.6</td>
<td>Fourth-Order Upwind Leapfrog Techniques for Scalar Advection</td>
<td>49</td>
</tr>
<tr>
<td>2.7</td>
<td>Phase Speed Characteristics for the Space and Time Extended Fourth-Order Upwind Leapfrog Techniques</td>
<td>51</td>
</tr>
<tr>
<td>2.8</td>
<td>Phase Speed Characteristics for Hermitian Fourth-Order Upwind Leapfrog Technique</td>
<td>52</td>
</tr>
<tr>
<td>2.9</td>
<td>Stability Range Demonstration</td>
<td>53</td>
</tr>
<tr>
<td>2.10</td>
<td>Fourth-Order Scheme Numerical Results to Model Scalar Advection Problem</td>
<td>54</td>
</tr>
<tr>
<td>3.1</td>
<td>Amplification Factors Plot for the Upwind Leapfrog Scheme to the Scalar Advection Equation with Source Term, ( \nu = 1/4 ), ( \theta = 2\pi/3 ) (( N = 3 ) cells-per-wave), and ( b\Delta t = 1/4 )</td>
<td>62</td>
</tr>
<tr>
<td>3.2</td>
<td>Upwind Leapfrog Techniques for the Spherical Wave Equation (( \Delta x = 1 ), ( t = 2000 ), ( \nu = \Delta t/\Delta r = 3/4 ))</td>
<td>64</td>
</tr>
<tr>
<td>3.3</td>
<td>Model Problem Time Evolution</td>
<td>69</td>
</tr>
<tr>
<td>3.4</td>
<td>Various Finite-Difference Techniques for Model Problem (( t = 99 ), ( a_\infty = 1/2 ), ( \nu = a_\infty \Delta t/\Delta x = 1/4 ))</td>
<td>70</td>
</tr>
<tr>
<td>3.5</td>
<td>The Upwind Leapfrog Scheme for Two Dimensional Advection</td>
<td>72</td>
</tr>
<tr>
<td>3.6</td>
<td>Coordinate Axis Aligned Non-Staggered Upwind Leapfrog Stencils</td>
<td>73</td>
</tr>
<tr>
<td>3.7</td>
<td>Mesh Oblique Non-Staggered Upwind Leapfrog Stencils</td>
<td>74</td>
</tr>
<tr>
<td>3.8</td>
<td>Staggered Upwind Leapfrog Stencils</td>
<td>75</td>
</tr>
<tr>
<td>3.9</td>
<td>Error Trends for the Non-Blended Coordinate Aligned Non-staggered Mesh Upwind Leapfrog Methods</td>
<td>77</td>
</tr>
</tbody>
</table>
4.11 Non-stationary Acoustics Positive x Axis Aligned Staggered Mesh Upwind Leapfrog Stencils .................................................. 116
4.12 Amplification Factors Plot for the Upwind Leapfrog Scheme with Compact ($\sigma_x = \sigma_y = 0$) Discretization ($M_x = 2/3, M_y = 1/3, v_x = 1/4, v_y = 3/8$) .................................................. 117
4.13 Amplification Factors Plot for the Upwind Leapfrog Scheme with Expanded ($\sigma_x = \sigma_y = 1$) Discretization $M_x = 2/3, M_y = 1/3, v_x = 1/4, v_y = 3/8$ .................................................. 118
4.14 Amplification Factors Plot for the Compromise Upwind Leapfrog Scheme ($\sigma_x = M_x$ and $\sigma_y = M_y$) Discretization $M_x = 2/3, M_y = 1/3, v_x = 1/4, v_y = 3/8$ .................................................. 119
4.15 Mesh Refinement Effect on Phase Speed Trends for the Second-Order Leapfrog Schemes for Two-Dimensional Non-Stationary Acoustics, $v_x = v_y = 1/4$, Background Flow Mach Number $M = 1/3$ Aligned 45° to Mesh .................................................. 120
4.16 Mach Number Effect on Phase Speed Trends for the Second-Order Leapfrog Schemes for Two-Dimensional Non-Stationary Acoustics, $v_x = v_y = 1/4$, Background Flow Aligned 45° to Mesh, and Coordinate Aligned Cells-per-Wavelength $N = 4$ .................................................. 121
4.17 Mesh Refinement Effect on Phase Speed Trends for the Higher-Resolution Upwind Leapfrog Schemes for Two-Dimensional Non-Stationary Acoustics, $v_x = v_y = 1/4$, Background Flow Mach Number $M = 1/3$ Aligned 45° to Mesh .................................................. 122
4.18 Mach Number Effect on Phase Speed Trends for the Higher-Resolution Upwind Leapfrog Schemes for Two-Dimensional Non-Stationary Acoustics, $v_x = v_y = 1/4$, Background Flow Aligned 45° to Mesh, and Coordinate Aligned Cells-per-Wavelength $N = 4$ .................................................. 123
4.19 Sample Pressure Fields for the Two-Dimensional Reflection Problem $50 \times 50$ Mesh .................................................. 125
4.20 Upwind Leapfrog Wall Reflection Results $v_x = v_y = 1/4$ .................................................. 126
4.21 Near Fourth-Order Upwind Leapfrog Wall Reflection Results $v_x = v_y = 1/4$ .................................................. 127
4.22 Fully Fourth-Order Upwind Leapfrog Wall Reflection Results $v_x = v_y = 1/4$ .................................................. 128
5.1 Upwind Leapfrog Stencil Arrangements for One-Dimensional Acoustic System .................................................. 131
5.2 Amplification Factors Plot for the Upwind Leapfrog Scheme to the One-Dimensional Acoustic Equation with Source Terms ($a \Delta t = 1/4, b \Delta t = 1/3, c \Delta t = 1/5, d \Delta t = 1/2, \nu = 1/4$, and $\theta = 2\pi/3$ ($N = 3$ cells-per-wave)) .................................................. 132
5.3 Amplification Factors Plot for the Upwind Leapfrog Scheme to the Two-Dimensional Acoustic Equations with a Constant Coefficient Source Term ($d_{12} = 1$, $\nu_x = 1/4$, $\nu_y = 3/8$, $\theta_x = 2\pi/5$ ($N_x = 5$), and $\theta_y = 3\pi/8$ ($N_y = 16/3$)).

5.4 Amplification Factors Plot for the Alternate Weightings Upwind Leapfrog Scheme to the Two-Dimensional Acoustic Equations with a Constant Coefficient Source Term ($d_{12} = 1$, $\nu_x = 1/4$, $\nu_y = 3/8$, $\theta_x = 2\pi/5$ ($N_x = 5$), and $\theta_y = 3\pi/8$ ($N_y = 16/3$)).
# LIST OF TABLES

2.1 Cells-per-Wavelength Requirements for Several Well Known Schemes to the Scalar Advection Equation ........................................ 41

2.2 Cells-per-Wavelength Requirements for Various Leapfrog Schemes to the Scalar Advection Equation ........................................ 51

3.1 Sizes of Symbolic Matrices that Must Be Inverted to Determine Modified Equations for Problems with Non-constant Coefficients .......... 58
LIST OF APPENDICES

A. GOVERNING EQUATIONS ............................................. 146

B. TWO-DIMENSIONAL ACOUSTIC SYSTEM GRADIENT APPROXIMATIONS ............................................. 150
CHAPTER I

INTRODUCTION

The rapid development of computers with ever increasing memories and speeds has enabled the numerical simulation of wave propagation problems of increasing complexity in fields such as acoustics and aeroacoustics, electro-magnetics, elasticity, and medical imaging. The computer however only comprises one aspect of numerical simulations to these problems. The numerical methods and algorithms behind these simulations also dictate whether such computations are efficient or even feasible. Most currently popular methods typically suffer from high dissipation and poor resolution for transient wave propagation problems. This dissertation presents an investigation into a relatively unknown scheme called the “upwind” leapfrog method which does not suffer from such problems at least for the case of scalar advection. I investigate how concepts behind the formulation of this scheme can be used to develop upwind leapfrog schemes for more complex forms of scalar advection. This includes scalar advection with source terms and non-constant coefficients and scalar advection in multi-dimensions. I also develop upwind leapfrog schemes for general systems of equations in multiple dimensions with a specific emphasis towards acoustics and aeroacoustics.
1.1 Initiative

Recent initiatives calling for quieter commercial and military aircraft have spawned research into the development of engineering tools that will enable engineers to predict noise characteristics of a preliminary aircraft configuration during the initial design stages. With the continuing success of computational fluid dynamic (CFD) methods where the emphasis has been the accurate and efficient simulation of complex steady-state flow fields, many researchers have begun investigating the possibility of accurately simulating unsteady flows. One particular area of interest is computational acoustics and aeroacoustics. Just as the motivation behind method development for steady flow is the elimination of the large time and expense of having to do physical testing such as wind-tunnel testing for aerospace vehicles, unsteady CFD method development is driven by similar motives. In the initial design stages, the possibility of a designer being able to predict the sound field via a computer simulation, and further being able to redesign and optimize before actually constructing a vehicle, has inspired the research in this dissertation. Even if the goal of being able to predict the entire sound field about an aircraft configuration is a ways off, current research may at least give researchers and engineers a better understanding of how sound waves behave in complex flow environments.

1.2 Governing and Model Equations

Problems in acoustics and aeroacoustics generally consist of one or more of the following three aspects. How sound is created, how it is propagated, and how it is transmitted between different media. The dissertation focuses on the development of numerical schemes for simulating the long range propagation of sound waves.

The general set of equations which govern the time dependent behavior of most
fluid flows are the Navier-Stokes equations. These non-linear equations are fully capable of capturing the physics of noise generation in addition to sound wave propagation for most non-rarefied gases. One problem in doing combined noise generation and propagation problems is that since there are several order of magnitude differences in the length and time scales between the two aspects, a single computational method is usually not well suited to handle both problems. Typically turbulent noise generation occurs on a length scale much smaller than the wavelength of the sound wave that it creates. As a consequence, the region of turbulence requires a much denser mesh than that of the far-field sound wave propagation region. Thus to do simulations of both phenomena simultaneously with an identical code would be computationally wasteful in that an unnecessarily large number of mesh points would be wasted in the wave propagation region. As a consequence, the current approach is to break up the domain of the problem into the two respective regions.

In the sound wave propagation portion of the problem, with the exception of regions where there might be high shears (i.e. boundary layers) the viscous terms in the Navier-Stokes equations which are responsible for noise generation are usually no longer a factor. Eliminating these terms leads to the Euler equations which can fully model the propagation of non-linear waves including traveling discontinuities (shocks). For most acoustics and aeroacoustics problems, the sound waves are typically several orders of magnitude less energetic than the mean flow quantities. As a consequence, shocks typically are not a factor in such flows. Thus one may simplify the equations even further to the classical aeroacoustic equations as described in the following.
1.2.1 Aeroacoustic Equations

To place my efforts in context, I first set out the governing equations that I would like ultimately to be able to solve. Under the approximation that the primitive variables of density $\rho$, velocity components $u_i$, and pressure $p$ are comprised of a steady “background” flow ($\rho_0(x_i), u_0(x_i)$, and $p_0(x_i)$) and transient portion ($\tilde{\rho}(x_i, t), \tilde{u}_i(x_i, t)$, and $\tilde{p}(x_i, t)$)

$$
\rho(x_i, t) = \rho_0(x_i) + \varepsilon \tilde{\rho}(x_i, t)
$$

$$
u_i(x_i, t) = u_0(x_i) + \varepsilon \tilde{u}_i(x_i, t)
$$

$$
p(x_i, t) = p_0(x_i) + \varepsilon \tilde{p}(x_i, t)
$$

a linearization to the governing equations can be made if $\varepsilon$ is small. For most sound waves, this is typically the case. For example, for sound pressure levels ranging from 60 dB (relatively quiet) to 120 dB (possible damage to hearing), $\varepsilon$ will range from being $O(10^{-6})$ to $O(10^{-3})$ which means a linearization should be acceptable. The linearized aeroacoustic equations that govern mass, momentum and energy for the transient variables may be expressed as (see Appendix A for full development)

$$
\frac{\partial \tilde{\rho}}{\partial t} + u_0 \frac{\partial \tilde{\rho}}{\partial x_i} + \rho_0 \frac{\partial \tilde{u}_i}{\partial x_i} = - \tilde{u}_i \frac{\partial \rho_0}{\partial x_i} - \tilde{\rho} \frac{\partial u_0}{\partial x_i}
$$

$$
\frac{\partial \tilde{u}_i}{\partial t} + u_0 \frac{\partial \tilde{u}_i}{\partial x_j} + \frac{1}{\rho_0} \frac{\partial \tilde{p}}{\partial x_i} = - \left( \tilde{u}_j + \frac{\rho_0 u_0}{\rho_0} \right) \frac{\partial u_0}{\partial x_j}
$$

$$
\frac{\partial \tilde{s}}{\partial t} + u_0 \frac{\partial \tilde{s}}{\partial x_j} = - u_0 \left( \frac{\tilde{\rho}}{\rho_0} \frac{\partial \rho_0}{\partial x_j} - \frac{\gamma \tilde{p}}{\rho_0} \frac{\partial \rho_0}{\partial x_j} \right)
$$

where I have defined the acoustic entropy $\tilde{s}$ as

$$
\tilde{s} = \tilde{p} - \frac{\gamma p_0}{\rho_0} \tilde{\rho} = \tilde{p} - a_0^2 \tilde{\rho}
$$
and \( a_0(x) \) is the local sound speed.

When there are no gradients in any of the mean flow quantities, the energy-entropy equation may be used to replace density in the continuity equation with pressure. In this situation, the system of equations is reduced by one, and the problem becomes one of pure acoustics, even though there may be a (uniform) background flow.

However, when there are gradients in the mean flow, the problem is then one of aeroacoustics. The energy-entropy equation is an advective-like equation, whereas the combination of the continuity and momentum equations describe both the transport of vorticity and the propagation of acoustic waves; they are both advective and wave-like in nature. In order to treat the full aeroacoustics equations, one needs to be able to devise schemes which can treat both the advective and wave-like phenomena. Furthermore, one needs to be able to do this for equations with non-constant coefficients and source terms.

### 1.2.2 Acoustic Equations

When there are no gradients in the background flow, the equations reduce to the pure acoustic equations

\[
\frac{\partial \tilde{\rho}}{\partial t} + u_{0i} \frac{\partial \tilde{\rho}}{\partial x_i} + \rho_0 \frac{\partial \tilde{u}_i}{\partial x_i} = 0 \quad (1.6)
\]

\[
\frac{\partial \tilde{u}_i}{\partial t} + u_{0j} \frac{\partial \tilde{u}_i}{\partial x_j} + \frac{1}{\rho_0} \frac{\partial \tilde{p}}{\partial x_i} = 0 \quad (1.7)
\]

\[
\frac{\partial \tilde{s}}{\partial t} + u_{0j} \frac{\partial \tilde{s}}{\partial x_j} = 0 \quad (1.8)
\]

where in this case, \( u_{0i}, \rho_0, \) and \( a_0 \) are constants. If there is no mechanism for creating variations of \( \tilde{s} \), and if \( \tilde{s} \) is initially constant everywhere as is typically the case for
pure acoustics, then $\dot{s}$ remains constant everywhere. If $\dot{s}$ is in this case zero, Equation (1.8) reduces to

$$\ddot{p} = a_0^2 \ddot{p}$$  \hfill (1.9)

the number of differential equations in the system is reduced by one. I have found that it is typically to easier to work with the equations when the acoustic density fluctuation $\dot{\rho}$ is replaced with the pressure fluctuation $\ddot{p}$

$$\frac{\partial \ddot{p}}{\partial t} + u_{0i} \frac{\partial \ddot{p}}{\partial x_i} + \rho_0 a_0^2 \frac{\partial \ddot{u}_i}{\partial x_i} = 0$$  \hfill (1.10)

$$\frac{\partial \ddot{u}_i}{\partial t} + u_{0j} \frac{\partial \ddot{u}_i}{\partial x_j} + \frac{1}{\rho_0} \frac{\partial \ddot{p}}{\partial x_i} = 0$$  \hfill (1.11)

If $u_{0i}$ is zero, the problem is one of stationary acoustics

$$\frac{\partial \ddot{p}}{\partial t} + \rho_0 a_0^2 \frac{\partial \ddot{u}_i}{\partial x_i} = 0$$  \hfill (1.12)

$$\frac{\partial \ddot{u}_i}{\partial t} + \frac{1}{\rho_0} \frac{\partial \ddot{p}}{\partial x_i} = 0$$  \hfill (1.13)

It possible to further reduce this system to a single variable by going to the second-order scalar wave equation. If one differentiates the first equation with respect to $t$ and second set of equations with respect to $x_i$ and subtracts, this yields the second-order wave equation for the pressure variable

$$\frac{\partial^2 \ddot{p}}{\partial t^2} - a_0^2 \frac{\partial^2 \ddot{p}}{\partial x_i^2} = 0$$  \hfill (1.14)

For the construction of most numerical schemes, it is however typically much easier to work with the system of equations.
1.2.3 Scalar Advection Equation

As was noted for the general aeroacoustics equations, the entropy equation is an advection equation of the form

\[ \frac{\partial u}{\partial t} + u_0 \frac{\partial u}{\partial x_i} = 0 \]  

(1.15)

with the addition of a source term. It can be shown that vorticity in aeroacoustics is also governed by such an equation. In the case of constant coefficients, this equation has an analytical solution of the form

\[ u(x_i, t) = u(x_i - u_0 t, t = 0) \]  

(1.16)

1.2.4 Time or Frequency Domain

One of the first questions one may ask is why does one need to solve the time dependent form when frequency domain approaches can be used. For example if I assume the solution has the form

\[
\begin{align*}
\tilde{\rho}(x_i, t) \\
u_i(x_i, t) \\
\tilde{p}(x_i, t)
\end{align*}
\right] = e^{iu_i t}
\begin{align*}
R(x_i) \\
U_i(x_i) \\
P(x_i)
\end{align*}
\]  

(1.17)

and substitute into any of the governing equation sets, I can change the character of the governing equations from being hyperbolic to elliptic. It is still necessary to solve a system of equations for the aeroacoustic equations, but in the case of a uniform background flow the system reduces to a single equation in \( P(x_i) \). In the case of stationary acoustics, the resulting equation is the Helmholtz equation

\[ \frac{\partial^2 \tilde{p}}{\partial x_i^2} + \frac{\omega^2}{c_0^2} \tilde{p} = 0 \]  

(1.18)

The frequency technique yields a field for only one particular frequency. For a problem where one might be concerned with a wide range of frequencies (for example,
normal human hearing extends from 20 Hz to 20 kHz), the time domain approach provides a better way of analyzing the acoustic field for such a large range of frequencies. The frequency of a noise source could be continually changed, and its immediate effects on the acoustic field would be known. Resonant frequencies and other features might also be picked up better in time-dependent simulations.

1.3 Numerical Methodology

In approaching the development of a numerical method for computational wave propagation problems, one generally has one of four different numerical techniques to choose from. These are finite-element, spectral, finite-volume, and finite-difference methods. In the following, some of the general advantages and disadvantages of each of these methods as applied to wave propagation problems are discussed.

The finite-element method is a technique in which the numerical solution is based upon the superposition of basis functions the coefficients of which are determined from some “weak form” of the governing equations. Advantages of the finite-element method are that irregular meshes are not a problem, and there is a rigorous mathematical framework in which to prove error bounds, stability, and convergence. A disadvantage lies in the fact that the underlying physics of a given problem is not easy to build into a given scheme (for example upwinding). Furthermore time and space discretizations are not easily coupled together as is desired for time-dependent wave propagation problems. However, this is beginning to change. Lowrie [LRvI.95] has recently investigated a space-time mesh approach using the discontinuous Galerkin technique with a focus towards long range wave propagation problems.

Spectral and pseudo-spectral methods are a special subset of finite-element methods in which the basis functions are high-order functions. The result is that spec-
tral methods have the highest level of accuracy. The only drawback is that this high accuracy can only be achieved on special pseudo-spectral meshes for general problems with arbitrary boundary conditions. The pseudo-spectral meshes tend to have very small spacing near boundaries which leads to very small time-stepping requirements. Upwinding and other physical attributes of a given problem cannot be built into the spectral discretizations and the time and spatial discretizations are usually treated independently. For time-dependent problems, typically Runge-Kutta or other lower-order time stepping methods must be used to enable scheme stability. Kopriva [Kop92, KK95] is currently one of the primary investigators of pseudo-spectral techniques for aeroacoustic wave propagation problems.

For finite-volume methods, the integral formulation of the governing conservation laws are discretized directly in the physical space. This ensures that the basic quantities of mass, momentum and energy remain conserved at the discrete level which makes the finite-volume method appropriate for flows with shocks. The method has the capability of being used on arbitrary meshes, and one may modify the resolution of the scheme by varying the rules and accuracy of the evaluation of the fluxes through the control volume surfaces. Although, the time and space discretizations are typically independent, it is through the use of concept of upwinding, which is achieved through the use of Riemann solvers in the evaluation of the fluxes, that help to unify the time and space discretizations in correctly modeling the true transient features of a flow.

In recent years, the essentially non-oscillatory method (ENO) [CHO86, SO87, SO88], has received considerable attention as a device to increase the resolution of finite-volume methods. However, one main drawback of ENO methods is that they typically lead to large computational stencils which present problems at computa-
tional boundaries. Atkins [LBA95, Atk95] has been one of the primary researchers investigating ENO methods for aeroacoustics type applications.

Finite-difference methods are designed to directly approximate the governing partial differential equations. The derivatives within the equations are approximated by algebraic differences (discretizations) of the dependent variables which are situated at discrete points within the computational domain of interest. A Taylor-series expansion of a discretization shows whether or not it is consistent with the derivative it is meant to represent along with how accurately, based on the width of the mesh, it represents the true derivative. This leads to a nice feature of finite-difference methods, namely that it is easy to construct a discretization to any desired level of accuracy just by incorporating, and weighting appropriately, a sufficient number of discrete mesh point values within the discretization. For time-dependent hyperbolic problems like advection and aeroacoustics where the physical nature of the analytic solution is known at the differential equation level, the discretizations can also be biased in a manner which best represents the known physics to achieve even better resolution. Upwind differencing, which may also be used in finite volume methods, is a primary example of this. Finally, since the linear aeroacoustic equations have no non-linear features like shocks which are more appropriately treated in the context of a finite-volume scheme, and since finite-difference methods are easy to construct, analyze, and work with, they currently are the most popular approach in this field. One typical drawback of finite-difference methods is that they are usually designed for, and are most accurate on uniform meshes. However, for the methods that I will be developing which must also appropriately model the non-constant coefficients which occur in the aeroacoustic equations, this turns out to not be too much of a problem since non-uniform spacing can be viewed as a form of non-constant coefficients. I
address this issue more in Chapter III.

Currently, there are two popular routes for devising high-resolution finite-difference methods for aeroacoustics. The first approach favors spatially extended computational stencils, and the second approach involves compact differencing methods.

In the spatially extended computational stencil approach, a scheme may be designed for either higher-order accuracy or optimized in various fashions to force the numerical dispersion relation to best follow the analytical dispersion relation. The upwind leapfrog based schemes I have developed take the approach of higher-order accuracy. Goodrich [Goo93] has also looked at high-order methods on spatially extended meshes except his methods are based on central as opposed to upwind differencing techniques.

Tam’s [TW92, TD94] extended stencil dispersion relation preserving (DRP) methods, Manthey’s [HHM95] low dissipation and dispersion Runge-Kutta schemes (LD-DRK), and Zingg’s [ZLJ93] optimized Runge-Kutta methods are examples of techniques where the schemes are designed to best follow the analytical dispersion relation in some optimized fashion. Although these techniques typically offer excellent resolution for high frequency data, most suffer from small time-step limitations.

Some examples of compact differencing methods are Lele’s [Le91] Padé approximation based techniques, and some recent optimized compact forms explored by Davis [Dav95] and Lee [KL96]. However, a typical drawback of compact methods is that they usually require expensive matrix inversions.

A new and different type approach is that of Chang [Cha95, CWC95]. Chang has created an interesting scheme where the solution gradients are carried along as extra dependent variables. Chang has been able to create a two-time level non-dissipative scheme using such data. His schemes are still formally second-order accurate; how-
ever, their non-dissipative nature for only a two time level scheme is unique.

1.3.1 Numerical Error Considerations

The objective for any numerical scheme which models a time-dependent hyperbolic problem like advection or aeroacoustics is to accurately and efficiently simulate the propagation of disturbances to the dependent variables over long periods of time within a given computational domain. Any given disturbance will be comprised of modes of varying frequencies which make up the spectrum of the disturbance, and any type of numerical method will typically not be able to exactly simulate the correct movement of each of these modes. At least one of two possible mechanisms will be responsible for any difference (numerical error) between the exact and numerical simulation.

The first mechanism which will generate error is the inability of a scheme to model the correct amplitude of a given mode as it propagates. This type of error is known as amplitude error. The second mechanism is inability of a scheme to move a given mode at the correct speed. This type of error is known as phase speed error. Figure 1.1 shows an illustration of these two types of error. To aid in the visualization of these errors, I have only drawn one wavelength of the mode. $A_e$ represents the amplitude error after the mode has propagated for a specific time, and $\lambda_e$ can be interpreted as the distance between where a mode arrives and where it should arrive after a specific time.

For the error analysis used throughout the rest of this dissertation, I use the following definitions to better represent each of the types of error. I define $A$ (amplitude error) as the difference between the exact and numerical amplitudes normalized by the exact amplification for a specific mode over one time step. For schemes applied
to linear constant coefficient equations on uniform grids, dispersion analysis, which
will be described in greater detail in the later chapters, can be used to come up with
values for the magnitude of the errors. A key parameter that comes out of dispersion
analysis is the scheme’s amplification factor $g$. The amplitude error expressed in
terms of $g$ turns out to be (where in this case the subscript $e$ denotes exact)

$$ A = \frac{A_e}{A_0} = \frac{|g_e| - |g|}{|g_e|} = 1 - \frac{|g|}{|g_e|} \tag{1.19} $$

I believe a better definition for amplitude error is to consider $A_e$ after a mode
has moved $R$ wavelengths. If $N$ represents the number of computational cells-per-
wavelength of the mode, and if $\nu$ (the CFL number) represents the ratio of the
time-step to the mesh width multiplied by the propagation speed ($\nu = a\Delta t/\Delta x$),
it can be shown that it takes $RN/\nu$ timesteps to move the mode $R$ wavelengths. I
define $D$ as the amplitude loss $A_e$ normalized by the original amplitude $A_0$ when a wave has moved $R$ number of wavelengths. This can be expressed as

$$D = 1 - \left( \frac{|g|}{|g_e|} \right)^{\frac{RN}{\nu}}$$

(1.20)

I think this definition is better in that one can get a sense of how much the amplitude will be off from the correct value after a mode has traveled a certain distance. It also avoids giving the misleading impression that small time steps imply accuracy. It is also easier to compare against the overall phase speed error $E$ which has a similar interpretation.

I define the overall phase error $E$ as the distance between where a mode arrives and where it should arrive normalized by the length of the wave itself after it has moved $R$ number of wavelengths. $E$ can be shown to be

$$E = \frac{\lambda}{\lambda_e} = R \left( \frac{\text{arg}(g)}{\text{arg}(g_e)} - 1 \right)$$

(1.21)

As will be shown in the subsequent chapters, most of the numerical schemes I have developed, which are based upon the upwind leapfrog method, have the nice inherent property of having zero dissipation error. Thus, my primary focus has been in improving the overall phase error properties. In the following section, I examine some of the payoffs one can expect from doing so. I have defined the amplitude errors since I do analyze and compare my methods to other schemes which do suffer from amplitude error.

1.3.2 Computational Cost Considerations

In this section, I conduct an examination of how certain parameters associated with the accuracy of a given scheme can affect the computational resources needed to solve a given problem. It can be shown that when the number of cells-per-wavelength
$N$ is sufficiently large, a scheme of order of accuracy $p$ will exhibit the following trend for the overall phase speed error $E$

$$E = R \left( \frac{\arg(g)}{\arg(g_e)} - 1 \right) = \begin{cases} \frac{Rk_E}{Np} & p \text{ even} \\ \frac{Rk_p}{Np+1} & p \text{ odd} \end{cases}$$ (1.22)

where $k_E$ is coefficient of the dispersive (even order) gradient term within the truncation. Similarly it can be shown that the amplitude error $A$ will exhibit the following trend

$$A = 1 - \frac{|g|}{|g_e|} = \begin{cases} \frac{Rk_D}{Np+2} & p \text{ even} \\ \frac{Rk_D}{Np+1} & p \text{ odd} \end{cases}$$ (1.23)

where $k_D$ is coefficient of the diffusive (odd order) gradient term within the truncation. The upwind leapfrog methods that have been developed in this thesis have the property of not suffering from any dissipation error. So in the following, I am just concerned with the phase speed error trends.

In comparing two different techniques, one may ask what the savings in computational resources might be if the order of accuracy is increased or the magnitude of the coefficient $k_E$ is reduced. Due to the odd/even order nature of the errors, comparing any two methods can become quite complex. For that reason, I consider a base scheme (denoted with the subscript 1) which is second-order accurate and typical of most methods in use today. In comparing this scheme to a different scheme (subscript 2), if both methods have the same level of overall phase error $E$, the following relations exist

$$\frac{k_{E2}}{N_2^p} = \frac{k_{E1}}{N_1^2} \quad p \text{ even}$$ (1.24)

$$\frac{k_{E2}}{N_2^{p+1}} = \frac{k_{E1}}{N_1^2} \quad p \text{ odd}$$ (1.25)
The first important and interesting thing to note from these relations is that I should get the same increase in performance by going with either a third or a fourth-order method. This turns out to be an unfortunate consequence for the upwind leapfrog method and other non-dissipative techniques. Namely, I must jump to a fourth-order variant before I see any improvement in performance. One might say that the best option is to then go to a third-order method. This is unfortunately not an option for the upwind leapfrog methods. Any third or odd-order variant will have some level of dissipation, and I strictly wish to develop non-dissipative techniques. Another question one may ask is given two schemes of the same order of accuracy, what affects does the ratio $k_{E2}/k_{E1}$ have on the required computational resources. From relation (1.24) (I just consider even-order accuracy)

$$\frac{N_2}{N_1} = \left( \frac{k_{E2}}{k_{E1}} \right)^{\frac{1}{p}}$$

As an example, consider if a new second-order scheme can be devised which has a factor of two improvement over an existing second-order scheme ($k_{E2}/k_{E1} = 1/2$). In this case, $\frac{N_2}{N_1} = 0.707$ which doesn’t seem like too much of a savings. However, in multiple spatial dimensions, the reduction in computational resources becomes much more significant. In $d$ dimensions (I consider $N$ to be the same in each coordinate direction), the ratio of computational resources is given by

$$\left( \frac{N_2}{N_1} \right)^d = \left( \frac{k_{E2}}{k_{E1}} \right)^{\frac{d}{p}}$$

which means if the same reduction can be made in three spatial dimensions, $\left( \frac{N_2}{N_1} \right)^3 = 0.353$, or less than one-half of the computational resources will be required for the improved scheme. The upwind leapfrog methods will be shown to exhibit this trait of having $k_{E2}/k_{E1} = 1/2$ over most other comparable methods. Furthermore, the overall number of computational operations which I denote as $M$ will have the following
This is due to the fact that the timestep can be increased on the coarser mesh. In three spatial dimensions $\left( \frac{M_2}{M_1} \right) = 1/4$ which is quite a savings considering the order of accuracy of the scheme did not even change.

The next question one might ask is how will increasing the order of accuracy effect the computational resources required. Considering $d$ spatial dimensions and $k_{E2} = k_{E1}$ for simplicity, it can be shown from Equation (1.24)\[ \left( \frac{N_2}{N_1} \right)^d = N_1^d \frac{k_{E2}}{k_{E1}} \]

where $r = p_2/p_1 = 1, 2, 3\ldots$. Unfortunately, since $\left( \frac{N_2}{N_1} \right)^d$ turns out to be a function of $N_1$, interpreting the effects of increasing the order of accuracy are difficult to ascertain unless considering specific values for $N_1$. Furthermore, one should also consider the added computational cost required in using the higher order method. Once a formula is known for the computational cost as a function of the order of accuracy, one might be able to determine the optimum accuracy to keep computational cost and resources at a minimum. Until that occurs, I have taken the approach of at least examining what the necessary requirements are to achieve the next higher level of accuracy, namely fourth-order, for the upwind leapfrog methods. Higher order methods will most likely be able to be developed in a similar fashion.

\subsection*{1.4 Objectives and Organization}

The primary objective of this dissertation has been to investigate whether high-resolution methods based on the relatively unknown upwind leapfrog method for simple scalar advection can be developed for the more complex system of equations
which governs aeroacoustics. The motivation has come from the fact that the upwind leapfrog method for scalar advection exhibits the very desirable traits of having zero dissipation and high resolution. A number of issues must be addressed in order to create an upwind leapfrog based scheme for aeroacoustics. These include non-constant coefficients, source terms, systems of equations, multiple dimensions, and fourth-order accuracy. This dissertation has been set up in the following arrangement to address each of these issues.

Beginning with Chapter II, I discuss classical methods for simple one-dimensional scalar advection and introduce the upwind leapfrog technique. I examine how schemes may be designed using polynomial interpolation or the equivalent differential equation, and I review dispersion analysis as a method to assess stability and accuracy. I also examine a few different options for developing fourth-order extensions of the upwind leapfrog scheme. In Chapter III, I address issues involved in applying the upwind leapfrog scheme and its higher-order variants to more general forms of the scalar advection equation. This includes non-constant coefficients, source terms, and multiple dimensions. In Chapter IV, I address multi-dimensional acoustics. I introduce the concepts of bicharacteristics and develop some second and fourth-order schemes in the context of the upwind leapfrog method for scalar advection. I also address non-stationary acoustics where a uniform background flow exists. In Chapter V, I examine developing upwind leapfrog based methods for aeroacoustic like systems of equations in one or more dimensions. Finally in Chapter VI, I present my conclusions and recommendations.

With the much appreciated aid of my advisor, I have currently worked out most of the issues that must be resolved in order to create a fully fourth-order upwind leapfrog based method for aeroacoustics. Unfortunately, the one last sticking point
has been treating the source terms for the system of equations in multiple dimensions.

I feel relatively confident in time this problem will be worked out. If not, some very accurate and efficient methods have still been developed for advection and pure acoustics type problems.
CHAPTER II

METHOD DEVELOPMENT FOR SCALAR ADVECTION

As noted in the Introduction, the aeroacoustic equations are comprised of both advective and wave-like portions. In this chapter, I focus on the design and analysis of numerical schemes that might be used for the advective portions of such flows. Specifically, I look at the case of simple one-dimensional scalar advection. In addition to examining a number of classical methods, I introduce and develop the upwind leapfrog scheme. Ideas used in the development of the upwind leapfrog technique for scalar advection will subsequently be used in developing analogous schemes for more complex forms of the scalar advection equation (Chapter III) and eventually the general acoustic (Chapter IV) and aeroacoustic (Chapter V) systems of equations.

I first illustrate the polynomial interpolation and equivalent differential equation techniques that can be used to design schemes for simple advection. I then show how such schemes can be analyzed for stability and accuracy using dispersion relation analysis. In these sections, I also develop the upwind leapfrog scheme and illustrate how it has much better numerical properties than most classical finite-difference schemes of the same order of accuracy. Finally, I examine some methods for developing even higher-order schemes based on the upwind leapfrog technique.
The scalar advection equation offers a simple model equation that may be used to illustrate the methods that will be subsequently used to design and analyze schemes for acoustics and aeroacoustics. In its simplest one-dimensional form it appears as

$$ u_t + au_x = 0 $$

(2.1)

where $u(x, t)$ is the unknown variable, $a$ is the advection speed, and $x$ and $t$ are the spatial and time coordinates. Scalar advection for a pure initial value problem has the solution

$$ u(x, t) = u(x - at, t = 0) = \Phi(x - at) $$

(2.2)

where $\Phi(x)$ is the initial data function. The solution states that in time $t$, the initial data $\Phi(x)$ will move a distance $at$ along the $x$ axis.

### 2.1 Design of the Upwind Leapfrog Scheme for Scalar Advection

In the following sections, I demonstrate three different approaches for developing the upwind leapfrog scheme for the simple scalar advection equation. The first method, which I call the polynomial interpolation technique, is a method which uses the known property of advection that the solution is constant along known characteristic paths. Using the solution values at previous time levels and mesh points near the spatial location where one desires an update as interpolants, one can create an interpolating polynomial of an order of accuracy depending on how many interpolants are employed. An update can then be created since one knows the point at which to evaluate the interpolating polynomial for the physics of advection. Computational stencils can then be designed to achieve the desired level of accuracy, and to also get the best arrangement of the interpolants. Unfortunately, the polynomial interpolation method can only be used for the pure scalar advection equation and not for
the scalar advection equation with source terms and/or non-constant coefficients. I do however present it here since it does give an indication of the features one might like to incorporate in the design of finite-difference schemes for any advective like equation.

The second design technique presented is the equivalent differential equation method. This technique, although not as insightful as the polynomial interpolation technique, does offer a way of creating finite-difference schemes for equations with source terms and/or non-constant coefficients. It also works for systems of equations such as aeroacoustics in multi-dimensions. Since this method is used extensively as a tool for designing higher-order numerical schemes for the more general acoustics and aeroacoustic equations, I give a thorough description of how this method is implemented. Due to the fact that I am developing schemes for equations with non-constant coefficients, a slightly different approach than the common “textbook” method for determining the modified equation is necessary.

Finally, I briefly explain a technique by which the upwind leapfrog scheme may be developed in the context of a conservative discretization.

### 2.1.1 Interpretation as Interpolation

Many of the classical numerical schemes for scalar advection can be constructed based on the idea of polynomial interpolation. For scalar advection, the independent variable \( u(x, t) \) is constant along characteristic paths \( dx/dt = a \). In the context of a discrete space and time domain \((x_i, t_i) = (i\Delta x, n\Delta t)\) with data known at every mesh point and \( m \leq n \) time level, one would like to determine the values of \( u(x, t) \) at the \( n + 1 \) time level. For any mesh point \( x_i \), the exact value for \( u(x, t) \) at the subsequent time level \((u_i^{n+1})\) is \( u(i\Delta x - a\Delta t, n\Delta t) \). Unfortunately, if \( u(i\Delta x - a\Delta t, n\Delta t) \) does not
lie on a mesh point, then one must find some approximation to its value. One way of doing this is to fit a $p$ order polynomial for $u(x, t)$ using $p+1$ mesh point values that lie about $u(i \Delta x - a \Delta t, n \Delta t)$. One can then use this polynomial to interpolate to an approximate value of $u$ at $x = i \Delta x - a \Delta t$. Several classical second-order schemes can be derived using this method. Figures 2.1(a,b) illustrate the computational stencils for both the Lax-Wendroff and second-order upwind schemes. For each of these

![Stencils](image)

Figure 2.1: Stencil Arrangements for Three Classical Second-Order Techniques along with Upwind Leapfrog Scheme for the Scalar Advection Equation

schemes, a second-order polynomial is fit to the data at $u_1$, $u_2$, and $u_3$ within each of the respective stencil arrangements. Based on these polynomials, the value for $u_i^{n+1}$
is approximated by an interpolation to \( u(i\Delta x - a\Delta t, n\Delta t) \). This yields the update for the Lax-Wendroff schemes as

\[
u^{n+1}_i = u^n_i - \frac{\nu}{2} \left( u^{n+1}_{i+1} - u^n_{i-1} \right) + \frac{\nu^2}{2} \left( u^{n+1}_{i+1} - 2u^n_i + u^n_{i-1} \right) \tag{2.3}
\]

and the second-order upwind scheme as

\[
u^{n+1}_i = u^n_i - \nu \left( u^n_i - u^n_{i-1} \right) + \frac{\nu(\nu - 1)}{2} \left( u^n_i - 2u^n_{i-1} + u^n_{i-2} \right) \tag{2.4}
\]

where \( \nu = a\Delta t/\Delta x \) is the Courant number.

Another well-known scheme which can be developed using polynomial interpolation is the standard leapfrog scheme (Figure 2.1(c)). Here, one also uses data at the \( n-1 \) time-level to construct the interpolating polynomial. Again, since the solution is constant along characteristic paths \( dx/dt = a \), the value of \( u^{n-1}_i \) extrapolated to the \( n \) time-level should be equivalent to \( u(i\Delta x + a\Delta t, n\Delta t) \). One can then use this value and its \( n \) time level \( x \) location as the third piece of information for the interpolating polynomial. Interpolating to \( u(i\Delta x - a\Delta t, n\Delta t) \) yields the standard leapfrog scheme as

\[
u^{n+1}_i = u^{n-1}_i - \nu \left( u^{n}_{i+1} - u^{n}_{i-1} \right) \tag{2.5}
\]

An interesting property of the standard leapfrog scheme is that it is neutrally stable as long as \( -1 \leq \nu \leq 1 \). The Courant number limitation is due to the fact that the domain of dependence of the governing equation must be contained within the domain of dependence of the discretization. The neutral stability means that as the solution is evolved in time, there will be no loss in its original amplitude. A way to convince oneself that the standard leapfrog scheme must be neutrally stable can be reasoned using the following argument. Suppose the scheme is not neutrally stable, but in fact dissipative to a degree that I will call \( |g| \). If the integration proceeds
backward in time, there will be growth of the solution at a rate $1/|g|$. But in fact the forward and reversed time schemes are identical, as can be seen from the stencil that represents the standard leapfrog scheme. If the stencil is “flipped” about the $n$ time level, it is exactly the same stencil as it was before. Therefore, as long as the scheme is stable, $|g|$ must be equal to one which implies the scheme is neutrally stable.

Another option to creating a second-order time reversible stencil is the upwind leapfrog scheme. Here one uses the data as $u^n_{i-1}$, $u^n_i$, and $u^{n-1}_{i-1}$. At the $n$ time level, $u^n_{i-1}$ should correspond to $u((i-1)\Delta x + a\Delta t, n\Delta t)$. Again using these three pieces of information to form a second-order interpolating polynomial and subsequently interpolating to $u(i\Delta x - a\Delta t, n\Delta t)$, one obtains the upwind leapfrog scheme

$$u^{n+1}_i = u^{n-1}_{i-1} - (2\nu - 1) \left( u^n_i - u^n_{i-1} \right)$$

(2.6)

For this scheme, the range of stable Courant numbers is $0 \leq \nu \leq 1$. Like the standard leapfrog method, if one looks at the scheme forwards or backwards in time, it is the same stencil in either sense which implies neutral stability and thus no dissipation error. Interestingly, even though the motivation for its stencil arrangement seems obvious, the upwind leapfrog scheme is currently a relatively unknown technique for the simple scalar advection equation. Most classic and current texts on numerical methods for scalar advection make no mention of it. In fact, I have not seen it appear in the literature until 1986 in a paper by Iserles [Ise86]. In the sections that follow, I show that the upwind leapfrog has many features which make it superior to all of the classical second-order schemes. One feature is the fact that it extends in space only one computational cell in width. This makes the scheme very easy to implement at computational boundaries. Also the compact spacing of the interpolants ($u_1$, $u_2$,
and $u_3$ typically leads to a much better approximation of $u(i\Delta x - a\Delta t, n\Delta t)$ which gives the scheme better resolution than other second-order methods.

2.1.2 Equivalent Differential Equation Approach

Another technique for designing schemes for scalar advection, along with more complex systems of equations, is the equivalent differential equation or modified equation approach. I illustrate this method since it can be used in the design of numerical schemes for more complex equations where the interpolation method is no longer possible. In the modified equation technique, an initial representation of the scheme is created by taking arbitrary weightings of finite-difference representations of various “legs” of the computational stencil that one has in mind for the discretization. Next, the modified equation is determined based upon these arbitrary weightings. The weightings are then set such that the resulting discretization is consistent with the governing differential equation and appropriate terms drop out of the truncation error to achieve the desired order of accuracy. I illustrate the idea with the Lax-Wendroff scheme; then present a generalized method that is applicable to any type of finite-difference scheme.

To begin with, I start with arbitrary weightings for each of the spatial legs of the Lax-Wendroff stencil as shown in Figure 2.1a. The discretization then proceeds as

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + c_1 \frac{u_{i+1}^n - u_i^n}{\Delta x} + c_2 \frac{u_i^n - u_i^{n-1}}{\Delta x} = 0$$  \hspace{1cm} (2.7)

The next step is to perform Taylor series expansions for all the terms within the discretization. To third-order, which is sufficient for the development of the second order Lax-Wendroff scheme

$$u_i^{n+1} = u_i^n + \Delta t u_t + \frac{(\Delta t)^2}{2} u_{tt} + O[(\Delta t)^3]$$  \hspace{1cm} (8a)
Substituting these expansions back into (2.7) yields

\[ u_t^n + \frac{\Delta t}{2} u_{tt} + c_1 \left( u_x + \frac{\Delta x}{2} u_{xx} \right) + c_2 \left( u_x - \frac{\Delta x}{2} u_{xx} \right) + \mathcal{O}[(\Delta x)^2, (\Delta t)^2] = 0 \] (9)

One can readily see that in order to be consistent with the scalar advection equation, \( c_1 + c_2 = a \). Specifying \( c_2 = a - c_1 \), the governing expansion becomes

\[ u_t + a u_x = (a - 2c_1) \frac{\Delta x}{2} u_{xx} - \frac{\Delta t}{2} u_{tt} + \mathcal{O}[(\Delta x)^2, (\Delta t)^2] \] (10)

where the right hand side is the truncation error. Setting \( c_1 = a/2 \) and thus eliminating the \( u_{xx} \) gradient yields the central difference forward Euler explicit method. This scheme is still however only first-order in time and is in fact an unstable discretization. Due to the fact that there is a time gradient in the truncation error of (10), one must determine the equivalent differential equation to ascertain how to set \( c_1 \) such that the resulting scheme is second-order in space and time.

The modified equation of a numerical scheme is the equivalent differential equation that the discretization satisfies (as in (10) above) with the exception that all the time gradients within the truncation error have been replaced with spatial gradients. In (10), the only term in the truncation error that needs to be replaced is the one containing \( u_{tt} \). One can find an expression for \( u_{tt} \) by taking the time derivative of Equation (10) itself.

\[ u_{tt} + a u_{xt} = (a - 2c_1) \frac{\Delta x}{2} u_{xxt} - \frac{\Delta t}{2} u_{ttt} + \mathcal{O}[(\Delta x)^2, (\Delta t)^2] \] (11)
If this expression for $u_{tt}$ is then substituted back into (10), the truncation error no longer has a term involving the $u_{tt}$ gradient. However, it now has a term involving the $u_{xt}$ gradient which must be subsequently removed

$$u_t + a u_x = (a - 2c_1) \frac{\Delta x}{2} u_{xx} + \frac{a \Delta t}{2} u_{xt} + \mathcal{O}[(\Delta x)^2, (\Delta t)^2]$$  \hspace{1cm} (12)

An expression for the $u_{xt}$ gradient can likewise be determined by differentiating (12) with respect to $x$.

$$u_{xt} + a u_{xxx} = (a - 2c_1) \frac{\Delta x}{2} u_{xxx} + \frac{a \Delta t}{2} u_{xxt} + \mathcal{O}[(\Delta x)^2, (\Delta t)^2]$$  \hspace{1cm} (13)

Substituting this expression for $u_{xt}$ back into (13), and the truncation error no longer has terms involving the $u_{tt}$ or $u_{xt}$ gradients. Thus the modified equation (to second-order) for the discretization (2,7) is

$$u_t + a u_x = \left[(a - 2c_1) \frac{\Delta x}{2} - \frac{a^2 \Delta t}{2}\right] u_{xx} + \mathcal{O}[(\Delta x)^2, (\Delta t)^2]$$  \hspace{1cm} (14)

By setting the coefficient of the $u_{xx}$ term in (14) equal to zero which implies that

$$c_1 = \frac{a}{2}(1 - \nu) \quad \text{and} \quad c_2 = \frac{a}{2}(1 + \nu)$$  \hspace{1cm} (15)

one forms the second-order Lax-Wendroff scheme (Equation (2,3)). It is important to note that the $u_{tt}$ gradient is eliminated first before the $u_{xt}$ gradient. It is not possible to obtain the modified equation unless the eliminations are carried out in that order. If I eliminate the $u_{xt}$ gradient first, during the process of then removing $u_{tt}$, the $u_{xt}$ gradient will “crop” back up within the truncation error. This will become more evident as I describe a more general method for obtaining modified equations in the following.

As one can see, the procedure just demonstrated will become rather lengthy especially as the complexity of the schemes and the order of the terms that one
wishes to see within the truncation error increases. I also make an important note here that this procedure cannot be used in the case of non-constant coefficients. I now outline a more general method for obtaining modified equations that can be used even in the case of non-constant coefficients which will be discussed in greater detail in Chapter III.

Beginning with an arbitrary discretization as was done for Equation (2.7) and substituting Taylor series expansions for each of the mesh point values within the stencil, one gets the general differential equation of the discretization

\[
\begin{align*}
& c_0 u + c_t u_t + c_x u_x + c_{tt} u_{tt} + c_{xt} u_{xt} + c_{xxx} u_{xxx} + \\
& + c_{ttt} u_{ttt} + c_{xxt} u_{xxt} + c_{xxxt} u_{xxxt} + c_{xxxx} u_{xxxx} + \\
& + c_{tttt} u_{tttt} + \cdots + \mathcal{O}[(\Delta x)^p, (\Delta t)^p] = 0
\end{align*}
\]

(16)

Here I have included the possibility of a source term and have considered the expansion up to order \(p - 1\). \(c_{m,n,t}\) represents the coefficient of the gradient \(u_{m,n,t}\) where \(m, n\) are integers. I will assume that the initial discretization is at least consistent with scalar advection which implies \(c_0 = 0, c_t = 1,\) and \(c_x = a\). I also note that all the coefficients in this case are constants

\[
\begin{align*}
& u_t + a u_x + c_t u_{tt} + c_x u_{xt} + c_{xx} u_{xx} + \\
& + c_{tt} u_{ttt} + c_{xt} u_{xxt} + c_{xxt} u_{xxxt} + c_{xxx} u_{xxxx} + \\
& + c_{ttt} u_{tttt} + \cdots + \mathcal{O}[(\Delta x)^p, (\Delta t)^p] = 0
\end{align*}
\]

(17)

This equation is considered the basic differential equation of the discretization. The goal once again is to eliminate all time gradient terms from second-order up to order \(p\) (i.e. \(u_{tt}, u_{xt}, u_{ttt}, u_{xxt}, \ldots\)) from the truncation error. I can find alternate equations for each of these gradients by differentiating (17) with respect to
the appropriate number of time and spatial derivatives. To determine the equation for the $u_{nl,mx}$ gradient where $n \geq 1$, $m \geq 0$, I operate on Equation (17) with $\frac{\partial^n}{\partial t^n} \frac{\partial^m}{\partial x^m}$. For example, to determine the equation for $u_{xtt}$, I would take the $\frac{\partial}{\partial t} \frac{\partial}{\partial x}$ partial derivative of (17). This yields

$$u_{xtt} + au_{xxt} + c_{xxtt}u_{xttt} + c_{xxxtt}u_{xxxtt} + c_{xxxxtt}u_{xxxxtt} +$$

$$+ c_{xxxxxtt}u_{xxxxxtt} + \cdots + O[(\Delta x)^p, (\Delta t)^p] = 0 \quad (18)$$

I have now included subscripts on each of the gradient coefficients to indicate which alternate equation is being considered. For example, the coefficient $c_{xxtt}$ represents the coefficient of the $u_{xxt}$ gradient in the alternate equation that has $u_{xtt}$ as the leading gradient.

By successively taking the appropriate time and space gradients of the basic differential equation (17), I can obtain all the alternate equations for each of the gradient terms with time components (again $u_{lt}$, $u_{xt}$, $u_{ltt}$, $u_{lttx}$, etc.). It turns out that all these equations when put together form a system of equations where the unknowns are the gradient terms with time components themselves (once again $u_{lt}$, $u_{xt}$, $u_{ltt}$, $u_{lttx}$, etc.). This system can be written in matrix form as

$$[C] \{u\}_t = - \{B\} \quad (19)$$

where the matrix $\{u\}_t$ represents the matrix of unknowns (gradient terms with time
that has terms involving only spatial gradients of \( u \) and \( f \) components)

\[
\{u\}_t = \begin{pmatrix}
    u_t \\
    u_{tt} \\
    u_{xt} \\
    u_{ttt} \\
    u_{xtt} \\
    u_{xxt} \\
    u_{tttt} \\
    \vdots
\end{pmatrix}
\]  

\( [C] \) is the coefficient matrix which has the form

\[
[C] = \begin{bmatrix}
    c_{t} & c_{tt} & c_{xt} & c_{ttt} & c_{xxt} & c_{tttt} & \cdots \\
    c_{tt} & c_{ttt} & c_{xtt} & c_{tttt} & c_{xxtt} & c_{ttttx} & \cdots \\
    c_{xt} & c_{xtt} & c_{xxt} & c_{xttt} & c_{xxttt} & c_{xtttx} & \cdots \\
    c_{ttt} & c_{tttt} & c_{xttt} & c_{ttttt} & c_{xxttt} & c_{tttttx} & \cdots \\
    c_{xxt} & c_{xxtt} & c_{xxtt} & c_{xxtttt} & c_{xxttttt} & c_{xxttttx} & \cdots \\
    \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots
\end{bmatrix}
\]  

and \( \{B\} \) is the vector that makes up the right hand side of the system of equations that has terms involving only spatial gradients of \( u \)

\[
\{B\} = \begin{bmatrix}
    c_{x} u_{x} + c_{xx} u_{xx} + c_{xxx} u_{xxx} + c_{xxxx} u_{xxxx} + \cdots \\
    c_{xt} u_{x} + c_{xxt} u_{xx} + c_{xxxt} u_{xxx} + c_{xxxxxt} u_{xxxxx} + \cdots \\
    c_{xtt} u_{x} + c_{xxtt} u_{xx} + c_{xxxtt} u_{xxx} + c_{xxxxxtt} u_{xxxxxx} + \cdots \\
    c_{xttt} u_{x} + c_{xxttt} u_{xx} + c_{xxxttt} u_{xxx} + c_{xxxxxttt} u_{xxxxxxx} + \cdots \\
    c_{xxtt} u_{x} + c_{xxttt} u_{xx} + c_{xxxttt} u_{xxx} + c_{xxxxxttt} u_{xxxxxxx} + \cdots \\
    \vdots
\end{bmatrix}
\]
Multiplying \{B\} by the inverse \[C\] yields the vector \(\{u\}_t\), the first element of which is the modified equation since it involves \(u_t\) and terms with only spatial gradients.

Now that the general procedure for using the matrix method of calculating the modified equation is known, as an example, I illustrate how the method works in determining the equivalent differential equation of the upwind leapfrog scheme to second-order. Performing a Taylor series expansion for the upwind leapfrog scheme \((2.6)\) up to second-order yields its differential form and truncation error as

Performing a Taylor series expansion for this scheme up to second-order yields its differential form and truncation error as

\[
u_t + a u_x = \frac{1}{2} (u_{xx} + u_{xt}) \Delta x - \frac{1}{2} \left( \frac{u_{xxt}}{2} + \frac{\nu^2 u_{ttt}}{3} + \frac{a u_{xxx}}{3} + \frac{\nu u_{xtt}}{2a} \right) \Delta x^2
\]

To obtain the modified equation to second-order, the matrix \(\{u\}_t\) has the following elements

\[
\{u\}_t = \begin{bmatrix}
u_t \\

u_{tt} \\

u_{xt} \\

u_{ttt} \\

u_{xtt} \\

u_{xrt}
\end{bmatrix}
\]

The total number of time gradient terms that must be eliminated to get a second-order modified equation is five which means solving a system of six equations. Based on the truncation error of \((23)\), and following the differentiation steps outlined before,
the coefficient matrix $[C]$ and vector $\{B\}$ for the upwind leapfrog scheme are

$$
[C] = \begin{bmatrix}
1 & 0 & -\frac{\Delta x}{2} & \frac{v^2 \Delta x^2}{6 a^2} & \frac{v \Delta x^2}{4 a} & \frac{\Delta x^2}{4} \\
0 & 1 & a & 0 & \frac{-\Delta x}{2} & \frac{-a \Delta x}{2} \\
0 & 0 & 1 & 0 & 0 & \frac{-\Delta x}{2} \\
0 & 0 & 0 & 1 & a & 0 \\
0 & 0 & 0 & 0 & 1 & a \\
0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
$$

(25)

$$
\{B\} = \begin{bmatrix}
u u_x - \frac{1}{2} a \Delta x u_{xx} + \frac{1}{6} a \Delta x^2 u_{xxx} \\
0 \\
0 \\
0 \\
0 \\
a \Delta x u_{xxx} \\
\end{bmatrix}
$$

(26)

One thing to notice about $[C]$ is that it has zero values in its lower triangle. Due to this fact, I could find the solution for $\{u\}_t$ without having to invert $[C]$ via backward substitution. This is the reason that I do not necessarily have to use the matrix method for finding the modified equations to constant coefficient equations. An example of this is at the beginning of this section when the modified equation for the Lax-Wendroff scheme was derived using simple substitutions. The reason, however, that I wish to use the matrix method, is that once there are non-constant coefficients, the lower triangle will no longer be all zeros.

Continuing with development of the modified equation for the upwind leapfrog
scheme, the inverse of $[C]$ is

$$[C]^{-1} = \begin{bmatrix}
1 & 0 & \frac{\Delta x}{2} & -\frac{\nu^2 \Delta x^2}{6 a^2} & \frac{\nu \Delta x^2 (2 \nu - 3)}{12 a} & -\frac{\nu^2 \Delta x^2}{6} & + \frac{\nu \Delta x^2}{4} \\
0 & 1 & -a & 0 & \frac{\Delta x}{2} & -a \\
0 & 0 & 1 & 0 & 0 & \frac{\Delta x}{2} \\
0 & 0 & 0 & 1 & -a & a^2 \\
0 & 0 & 0 & 0 & 1 & -a \\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

(27)

Multiplying the first row of $[C]^{-1}$ by the column vector $\{B\}$ yields the modified equation. The following list gives the modified equations for the upwind leapfrog scheme as well as the other second-order methods that have been discussed up to fifth or sixth order using the matrix method

**Upwind Leapfrog**

$$u_t + a u_x = \frac{a \Delta x^2 (2 \nu - 1) (\nu - 1) u_{xxx}}{12} - \frac{a \Delta x^4 (\nu - 1) (3 \nu - 1) (2 \nu - 1) (3 \nu - 2) u_{xxxxx}}{240} + O[(\Delta x)^6] \quad (28)$$

**Standard Leapfrog**

$$u_t + a u_x = \frac{a \Delta x^2 (\nu - 1) (\nu + 1) u_{xxx}}{6} - \frac{a \Delta x^4 (\nu - 1) (3 \nu - 1) (3 \nu + 1) (\nu + 1) u_{xxxx}}{120} + O[(\Delta x)^6] \quad (29)$$

**Lax-Wendroff**

$$u_t + a u_x = \frac{a \Delta x^2 (\nu - 1) (\nu + 1) u_{xxx}}{6} + \frac{a \nu \Delta x^3 (\nu - 1) (\nu + 1) u_{xxxx}}{8} + \frac{a \Delta x^4 (\nu - 1) (\nu + 1) (6 \nu^2 + 1) u_{xxxxx}}{120} + O[(\Delta x)^5] \quad (30)$$
Second-Order Upwind

\[ u_t + a u_x = \frac{a \Delta x^2 (\nu - 1) (\nu - 2) u_{xxxx}}{6} + \frac{a \Delta x^3 (\nu - 2) (\nu - 1)^2 u_{xxxxx}}{8} + \frac{a \Delta x^4 (\nu - 1) (\nu - 2) (6\nu^2 - 12\nu + 7) u_{xxxxxxx}}{120} + \mathcal{O}[(\Delta x)^5] \]  

Fromm’s Scheme

\[ u_t + a u_x = \frac{a \Delta x^2 (2\nu - 1) (\nu - 1) u_{xxx}}{12} + \frac{a \Delta x^3 (\nu - 1) (\nu^2 - \nu + 1) u_{xxxx}}{8} + \frac{a \Delta x^4 (\nu - 1) (2\nu - 1) (6\nu^2 - 6\nu + 13) u_{xxxxx}}{240} + \mathcal{O}((\Delta x)^5) \]  

A few things should be noted for each of the modified equations. First, it can be seen that the leapfrog and upwind leapfrog schemes have no even order gradient terms within their respective truncation errors. Even ordered gradient terms are diffusive (like viscosity) and imply damping. As was noted earlier in the design of the leapfrog and upwind leapfrog schemes, their stencil arrangements imply neutral stability, and thus the modified equations prove this is so. It can be seen that the Lax-Wendroff, second-order upwind, and Fromm’s scheme [Fro68] (which is an average of the Lax-Wendroff and second-order upwind schemes) each have fourth-order spatial gradient terms. This implies that these schemes have some level of dissipation, the magnitude of which will be derived in the section on dispersion analysis. It is also interesting to note that the coefficient of the leading error term in the truncation error of upwind leapfrog scheme is a factor of two smaller than that of the standard leapfrog scheme. Warming and Hyett [WH74] have shown that

\[ E% = \frac{100}{\nu} \sum_{n=1}^{\infty} (\frac{2\pi}{N})^{2n} c_{2n+1} \]  

where \( c_{2n+1} \) is the coefficient of the \( u_{(2n+1)x} \) term within the truncation error of the modified equation. For sufficiently large \( N \), one only needs to retain the first term.
of (33). In this case the ratio of the phase speed errors for the upwind leapfrog (ulf) and standard leapfrog schemes (slf) is given by

\[
\frac{E_{ulf}}{E_{slf}} = \frac{1}{2} \frac{2\nu - 1}{\nu + 1}
\]

(34)

For \(0 \leq \nu \leq 1\),

\[
\left| \frac{E_{ulf}}{E_{slf}} \right| \leq 1/2
\]

(35)

Based upon this type of comparison, which again only holds for sufficiently large \(N\), the upwind leapfrog scheme has at least twice the phase speed accuracy as each of the other schemes except for Fromm’s in which it is same. It is important to note however that Fromm’s scheme has a large stencil which spans three computational cells, and it is dissipative to a degree that will be shown in the next section.

2.1.3 A Conservative Formulation

I now present an interesting way to develop the upwind leapfrog scheme in the context of a finite-volume formulation. This may be done by enforcing conservation over two time-levels. The first step is to cast the simple advection equation into a conservation law expressed as

\[
\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = 0 \quad \text{where} \quad f = au
\]

(36)

Based upon the control volume shown in Figure 2.2 conservation requires

\[
\frac{u_N - u_S}{2\Delta t} + \frac{f_E - f_W}{\Delta x} = 0
\]

(37)

The next step is to determine the fluxes \(f_E\) and \(f_W\) such that the resulting discretization turns out to be the upwind leapfrog scheme. For the scalar advection equation, this means the fluxes need to have form

\[
f_E = au_0 + a \frac{u_S - u_0}{2\nu} \quad f_W = au_1 + a \frac{u_2 - u_1}{2\nu}
\]

(38)
As can be seen from the fluxes, the scheme uses a first-order upwind representation of \( u_x \) with a second-order correction. Since \( u \) is constant along the characteristic paths \( dx/dt = a, u_S = u(i \Delta x + a \Delta t, n \Delta t) \), \( u_2 = u((i - 1) \Delta x + a \Delta t, n \Delta t) \) and \( a \Delta t \) can be considered the distance between these two values at a given time-level. Inserting these relations back into (38) yields for \( f_E \)

\[
f_E = a u_0 + \frac{a u(i \Delta x + a \Delta t, n \Delta t) - a u(i \Delta x, n \Delta t)}{a \Delta t} \frac{\Delta x}{2}
\]

As can be seen more clearly now, the correction term is an approximation of the gradient \( f_x \) multiplied by the distance \( \Delta x / 2 \) to give a better estimation of \( f_E \). The troubling aspect of this discretization is the \( 1/\nu \) dependency in the correction term. Another problem is the case of a non-uniform mesh in which instance, \( \nu \) is no longer constant. When the complete update (Equation (37)) is put together on uniform mesh, the point \( u_S \) does not end up appearing in the discretization. This will not be the case however on a non-uniform mesh. The value of \( u_S \) begins to play a role in the update, and the scheme no longer remains neutrally stable. Future investigations
might look into methods for adding a limiter to the above formulation.

2.2 Error Analysis

I now briefly review some analytical techniques that enable one to explicitly determine the numerical properties of many of the finite-difference schemes that I will be developing. These techniques can be used to determine the magnitudes of dissipation and dispersion error for any finite-difference scheme that models a differential equation or system of equations with constant coefficients on a uniform mesh. In the case of non-constant coefficients or non-uniform meshes, numerical simulations must be compared against exact solutions in order to discern the levels of error. This situation will be discussed more in Chapter III. Here, I also present some sample results for the upwind-leapfrog scheme against some classical schemes in order to demonstrate its capabilities.

2.2.1 Dispersion Relations

Fourier/von Neumann or dispersion relation analysis are two approaches for assessing the stability and accuracy of linear numerical schemes. The constraints are that the scheme must be considered over a uniform mesh in modeling equations with constant coefficients. Effects of alternative discretizations at computational boundaries also cannot be taken into account. The methods are very similar, and produce the same results. I find the dispersion relation method a little easier work with.

In dispersion analysis, the solution is assumed to be of the form

$$u(x, t) = e^{i(\omega t - \kappa x)}$$

where $\kappa$ is the spatial frequency of the data, and $\omega$ is the temporal frequency. $\kappa$ and
\( \omega \) are related to the Fourier angles \( \theta \) and \( \phi \) via

\[
\kappa = \frac{\theta}{\Delta x} = \frac{2\pi}{N\Delta x} \quad \text{and} \quad \kappa = \frac{\phi}{\Delta t}
\]  

(41)

where I have also shown how the number of cells-per-wavelength \( N \) is related to these angles. Substituting (40) and (41) into the scalar advection equation (2.1) yields the analytical relation between the temporal and spatial Fourier angles as

\[
\omega_c = a\kappa \quad \text{or} \quad \phi_c = \nu \theta
\]  

(42)

where the subscript \( e \) denotes exact.

To obtain the dispersion relation for a numerical scheme, one simply inserts (40) into the scheme’s discretization. For the standard leapfrog scheme (2.5), one gets

\[
\frac{e^{i[\omega(t+\Delta t)-\kappa x]} - e^{i[\omega(t-\Delta t)-\kappa x]}}{2\Delta t} + a \frac{e^{i[\omega(t-\kappa(x+\Delta x)]} - e^{i[\omega(t-\kappa(x-\Delta x)]}}{2\Delta x} = 0
\]  

(43)

Dividing through by \( e^{i[\omega t-\kappa x]} \)

\[
\frac{e^{i\omega \Delta t} - e^{-i\omega \Delta t}}{2\Delta t} + a \frac{e^{-i\kappa \Delta x} - e^{i\kappa \Delta x}}{2\Delta x} = 0
\]  

(44)

and by further using the trigonometric relation \( e^{i\beta} - e^{-i\beta} = 2i \sin(\beta) \), the dispersion relation for the standard leapfrog scheme becomes

\[
\sin(\omega \Delta t) = \nu \sin(\kappa \Delta x)
\]  

(45)

or in terms of the Fourier angles from relation (41)

\[
\phi = \arcsin(\nu \sin(\theta)) = \arcsin(\nu \sin \left( \frac{2\pi}{N} \right))
\]  

(46)

Following the same procedure, one obtains the following dispersion relations for the upwind leapfrog scheme

\[
\phi = \frac{\theta}{2} + \arcsin \left[ (1 - 2\nu) \sin \left( \frac{\theta}{2} \right) \right] = \frac{\pi}{N} + \arcsin \left[ (1 - 2\nu) \sin \left( \frac{\pi}{N} \right) \right]
\]  

(47)
The first thing to note is that as long as $-1 \leq \nu \leq 1$ in the case of the standard leapfrog scheme, and $-1 \leq 1 - 2\nu \leq 1$ (which implies $0 \leq \nu \leq 1$) in the case of the upwind leapfrog scheme, $\phi$ will be real valued. If $\phi$ is written as

$$\phi = \phi_R + I \phi_I$$

(48)

where the subscripts $R$ and $I$ denote the real and imaginary components of $\phi$, from Equation (40), the solution for $u(x, t)$ will have the form

$$u(x, t) = e^{-\phi_I \frac{x}{\Delta x}} e^{I(\nu \frac{x}{\Delta x} + \phi_R \frac{t}{\Delta t})}$$

(49)

Thus if $\phi$ has a nonzero imaginary component $\phi_I$ (which is the case for unstable or dissipative schemes), then the amplification of Equation (40) will be modified by a factor of $e^{\phi_I \frac{x}{\Delta x}}$. The amplification of a scheme is defined as

$$g = \frac{u_{i}^{n+1}}{u_{i}^{n}} = e^{I\phi}$$

(50)

which upon considering the data as in (40) gives

$$g = e^{-\phi_I} e^{I\phi_R}$$

(51)

the exact value of which for the scalar advection equation is

$$g_e = e^{I\nu \theta}$$

(52)

The dissipation and dispersion errors as defined in Section 1.3.2 (Equations (1.20) and (1.21)) for the one-dimensional scalar advection equation are thus given by

$$D = 1 - \left(\frac{|g|}{|g_e|}\right)^{\frac{\Delta x}{\nu}} = 1 - \left(e^{-\phi_I} \frac{\Delta x}{\nu}\right)$$

(53)

$$E = R \left(\frac{\arg(g)}{\arg(g_e)} - 1\right) = R \left(\frac{\phi_R}{\nu \theta} - 1\right)$$

(54)
Table 2.1: Cells-per-Wavelength Requirements for Several Well Known Schemes to the Scalar Advection Equation

As mentioned in the Introduction, an important question one may ask concerning the computational efficiency of a scheme, is what is the least number of cells-per-wavelength needed to maintain a fixed level of dissipation or phase speed error. Table 2.1 shows the trend for several well known schemes for scalar advection when $D(R = 1)$ and $E(R = 1)$ are limited to just one percent. Since most schemes have zero error for Courant numbers of zero, and/or one (the upwind leapfrog scheme also has zero error for $\nu = 1/2$), I have examined Courant numbers of $1/4$ and $3/4$ which tend to be more difficult. As can be seen, the first-order upwind and Lax’s schemes are extremely dissipative. Several hundred or even several thousand computational cells are required just to move the initial data the distance of one wavelength and preserve 99 percent of amplitude. This is mainly due to the fact that the first term of order $\Delta x$ in the truncation error consists of the diffusive $u_{xx}$ gradient. Clearly such schemes are not practical for transient simulations. The second-order schemes fare much better. Lax-Wendroff, second-order upwind, Fromm’s, and the standard leapfrog schemes can all preserve both the dissipation and phase speed errors to one percent with around 25 cells-per-wavelength for the Courant numbers I have
examined. Even the third-order upwind scheme requires 22 cell-per-wavelength to limit the dissipation error to one percent. The exciting result is that the upwind leapfrog scheme (which is still only second-order) can meet the error limitations with as few as 11 cells-per-wavelength. It also never suffers from dissipation error since it is neutrally stable like the standard leapfrog scheme.

Figures 2.3 shows the phase error trends for the standard and upwind leapfrog schemes. Shown are contours for various levels of phase speed error $E$ in a plane that shows the the number of cells-per-wavelength $N$ versus Courant number $\nu$. It is not necessary to consider similar plots for $D$ since neither of the schemes suffers from dissipation error. A few interesting features are important to point out. For the range of valid Courant numbers, the upwind leapfrog scheme suffers much less phase error than the standard leapfrog scheme. One nice feature is that it is exact for a Courant number of $1/2$. It also tends to be more accurate in the Courant number range $1/2 \leq \nu \leq 1$ than for $0 \leq \nu \leq 1/2$.

The primary advantage of the standard leapfrog method over the upwind leapfrog scheme is that it may be used for Courant numbers less than 0, that is for data moving in either direction. In the following chapters, this turns out to be one of the complications (as is the case for most upwind schemes) when applying the upwind leapfrog scheme to systems of equations where the advection or wave directions vary or change sign.

2.2.2 Group Velocity Considerations

Using the dispersion relation technique for stability and accuracy analysis, one may also determine the group speed properties of a given scheme. It can be shown [J78] that the group velocity is the speed at which the energy of a wave will travel. When
the initial data consists of varying frequencies, its overall movement will tend to
proceed at the group velocity. The group velocity is defined as the derivative of $\omega(\kappa)$
with respect to $\kappa$.

$$v_g = \frac{d\omega}{d\kappa} = \frac{d\phi}{d\theta} \frac{\Delta x}{\Delta t}$$

(55)
The analytical group speed velocity for the scalar advection equation is the advection speed \( v_g = a \). Using Equations (45) and (47) one can show that the numerical group speed velocity for the standard leapfrog scheme is

\[
v_g = \frac{a \cos(\theta)}{\sqrt{1 - \nu^2 \sin^2(\theta)}}
\]

and for the upwind leapfrog scheme

\[
v_g = \frac{a}{2\nu} \left[ 1 - \frac{(1 - 2\nu) \cos\left(\frac{\theta}{2}\right)}{\sqrt{1 - (1 - 2\nu)^2 \sin^2\left(\frac{\theta}{2}\right)}} \right]
\]

Several interesting features that are not readily apparent from the phase speed relations can be seen by examining the group speeds for each of these schemes at some specific Fourier angles. At the highest frequency resolvable by the mesh, \( \theta = \pi \), (two cells-per-wavelength), the group speed for the standard leapfrog scheme turns out to be \( v_g = -a \). This means a wave packet which consists of frequencies close to \( \theta = \pi \) will actually move at the correct speed but in the wrong direction. For \( \theta = \pi/2 \) (four cells-per-wavelength), the numerical group speed is zero which implies that such a wave packet will not move at all. In the next section, I illustrate this phenomenon occurring in a sample problem. For high frequency data, the standard leapfrog scheme has rather undesirable properties. Things are a little better for the upwind leapfrog scheme although it does have a slight problem that should be addressed.

In the case of the upwind leapfrog scheme, at the highest frequency resolvable by the mesh \( \theta = \pi \) the group speed is

\[
v_g = \frac{a}{2\nu}
\]

Thus for Courant numbers less then 1/2, the group speed is faster than the advection speed, and for Courant numbers greater than 1/2 the group speed is slower than the
advection speed. Although the upwind leapfrog scheme cannot match the exact
group velocity, it does at least move the data in the correct direction. The slight
problem that the upwind leapfrog suffers from is poor group speed properties for
small Courant numbers. This is due to the group velocity’s $1/\nu$ dependency. If one
defines group velocity error as

$$E_g = R \left( \frac{v_g - a}{a} \right) = R \left( \frac{v_g}{a} - 1 \right)$$

which is conceptually the same as the overall phase speed error $E$, then the dis-
tance between where a wave packet end ups and where it was supposed to end up
normalized by the characteristic wavelength ($N = 2$ in this case) after moving $R$
wavelengths becomes infinite as $\nu$ approaches zero for the upwind leapfrog scheme.
To sum up, the upwind leapfrog scheme has a problem advecting the two cell-per-
wavelength data as the Courant number goes to zero. If $\nu$ does not have to be
extremely small for a particular simulation, the upwind leapfrog still exhibits much
better characteristics than other finite-difference schemes in its class. Also, for lower
frequency data, the singularity does not exist. For example, in the limit as $\nu$ goes to
zero, $v_g(\theta = \pi/2) = 2a$, $v_g(\theta = \pi/3) = 4a/3$.

2.2.3 Performance Comparisons

In this section, I illustrate the properties of some of the second-order schemes by
modeling scalar advection with unit wave speed ($u_t + u_x = 0$) with initial data

$$u(x, 0) = \sin(50\pi x) \exp \left[ -100(x - \frac{1}{2})^2 \right]$$

This initial condition represents a sine wave modulated by a Gaussian envelope.
Using a uniform mesh with $\Delta x = 1/100$ unit which corresponds to four cells-per-
wavelength for the initial data, the data is advected a distance of 25 wavelengths.
With periodic boundary conditions, this should bring the initial condition back to its initial location. A time-step is taken such that the Courant number is $\nu = 1/4$. Figure 2.4 shows numerical results to this problem using the Lax-Wendroff, standard, and upwind leapfrog techniques. The solid lines represent the envelope of the exact solution, and the dashed lines represent the numerical results. All the leapfrog methods use the Lax-Wendroff method for the initial time-step. Granted this introduces a bit of dissipation in the first step. However, since there are 99 more time-step taken in this simulation, the effects of the first time step are very small in regard to the overall simulation. The dissipative nature of the Lax-Wendroff can readily be seen from Figure 2.4a. After moving 25 wavelengths, the initial condition has been almost completely attenuated. This example readily illustrates the destructive effects of scheme which has any level of dissipation. It should be noted that if one looks back to Table 2.1, the cell-per-wavelength requirements for maintaining dissipation and phase error to the same low level are roughly the same. But as can be seen from this figure, the dissipation error appears to have a much more destructive effect on coarse meshes in that the phase error can no longer be visually assessed at the scale in which the figure is drawn. Any of the other second-order schemes that have
dissipation error also produce similar results. This has been one of the main reasons for looking to non-dissipative schemes for long range wave propagation problems in general. The non-dissipative schemes give much better, although not ideal results to this problem.

Figure 2.4b shows the result for the standard leapfrog scheme. As mentioned in the previous section, when the characteristic (or primary) frequency of the wave packet corresponds to four cells-per-wavelength, the group velocity for the standard leapfrog scheme is zero. This result is readily demonstrated in the figure. Although the solution appears to be in the correct position, the data has not even moved from the initial time. The inset figure is used to help to show this effect more clearly. There is some spreading of the initial data since it is composed of varying frequencies, but as predicted by group velocity, the overall packet does not move at all. Although the standard leapfrog technique satisfies the non-dissipative criterion, it is this undesirable group speed property that has been one motivation to seek alternatives to the standard leapfrog technique.

Finally Figure 2.4c shows the result for the upwind leapfrog scheme. From the group speed relation for the upwind leapfrog scheme (Equation 57), for $N = 4$ cells-per-wave ($\theta = \pi/2$) and $\nu = 1/4$, the group velocity is $2a(1 - \sqrt{7}/7)$. The overall group speed error as defined by Equation 59 is $E_g = 6.107$. Multiplying this by the characteristic wavelength should give the distance between where the wave packet ended up in the simulation, and where it should have ended up. The wavelength is $1/25$ units, so the distance between the peaks of the numerical and exact solutions should be approximately $0.25$ units. Examining Figure 2.4c, one can readily see that this appears to be the case. Once again, like the standard leapfrog scheme, the data has spread a bit, but it appears to have been kept together much better. Most
importantly, the simulation has been able to move the data much closer to the exact advection speed without any of destructive diffusion exhibited by the dissipative second-order schemes. Of course, I do not regard (c) as an acceptable solution, but by using these very coarse meshes, I exaggerate the faults of the alternatives. It turns out that even better numerical results can be achieved without a significant increase in computational effort by developing some higher-order variations of the upwind leapfrog scheme. This is done in the following section.

2.3 Higher-Order Extensions

In this section, I present a few methods for developing fourth-order schemes for the scalar advection based on the concepts behind the second-order upwind leapfrog technique. If one continues to consider the schemes for scalar advection as interpolation routines, there are several options for gaining higher-order accuracy. I am going to stipulate zero dissipation for the schemes due to the destructive effects of diffusion as was shown in the previous section. The new scheme’s stencils must then still maintain the property of time reversibility or symmetry.

2.3.1 Fourth-Order Upwind Leapfrogs

To get fourth-order accuracy, two additional pieces of data are necessary to fit a unique fourth-order polynomial. The two extra pieces of data may be obtained in one of three different ways. First, the stencil may be stretched out in space. This is shown in Figure 2.5a. Another way of achieving higher accuracy is to stretch the stencil in time. By keeping the data from the \( n - 2 \) time level, one can devise another fourth-order scheme based on the stencil in Figure 2.5b. Finally by updating and carrying the slopes as new variables (a Hermitian approach) one can devise a fourth-order scheme based on the stencils in Figure 2.6. Even higher accuracy can
of course be gained by combining these strategies.

Figure 2.5: Fourth-Order Upwind Leapfrog Techniques for Scalar Advection

Figure 2.6: Fourth-Order Upwind Leapfrog Techniques for Scalar Advection
The scheme corresponding to the extension of the stencil in space is

\[
 u_{i+1}^{n+1} = u_{i}^{n+1} + \frac{(\nu + 1)(2\nu - 1)(\nu - 2)}{2} (u_{i}^{n} - u_{i-2}^{n}) - \frac{\nu (\nu - 1)(2\nu - 1)}{6} (u_{i+1}^{n} - u_{i-2}^{n})
\]

and the scheme corresponding to the extension of the stencil in time is

\[
 u_{i+1}^{n+1} = u_{i-1}^{n+1} - 2(3\nu - 1) (u_{i}^{n} - u_{i-1}^{n-1}) - \frac{(3\nu - 1)(2\nu - 1)}{\nu + 1} (u_{i}^{n-1} - u_{i-1}^{n-1})
\]

When the Hermitian technique is employed, a total of six pieces of data are available for the polynomial interpolation \((u_{i}^{n}, u_{i-1}^{n}, u_{i-1}^{n-1}, u_{x_{i}}^{n}, u_{x_{i-1}}^{n}, u_{x_{i-1}}^{n-1})\). Using all six pieces of data yields a fifth order scheme for scalar advection. However, the resulting scheme is unstable, containing singular terms proportional to \(\nu^{-1}\) and \((1 - \nu)^{-1}\).

A non-dissipative fourth-order scheme results from ignoring \(u_{x_{i}}^{n+1}\) when predicting \(u_{i+1}^{n+1}\), and ignoring \(u_{i-1}^{n+1}\) when predicting \(u_{x_{i}}^{n+1}\). The scheme is, abbreviating \(u_{x} \Delta x\) by \(s\),

\[
 u_{i}^{n+1} = (2\nu - 1) (2\nu^2 - 2\nu - 1) (u_{i}^{n} - u_{i-1}^{n+1}) - \nu (2\nu - 1) (\nu - 1) (s_{i}^{n} + s_{i-1}^{n}) + u_{i-1}^{n-1} 
\]

\[
 s_{i}^{n+1} = -12\nu (\nu - 1) (u_{i}^{n} - u_{i-1}^{n}) + (6\nu^2 - 6\nu + 1) (s_{i}^{n} + s_{i-1}^{n}) - s_{i-1}^{n-1} 
\]

Like the second-order scheme results presented in Table 2.1, Table 2.2 shows the number of cells-per-wavelength \(N\) required for each of the fourth-order upwind leapfrog schemes to be within one percent accuracy. Figures 2.7 and 2.8 show phase speed error \(E\) (in percent) in the \((N,\nu)\) plane. As can be seen from the figures and the
<table>
<thead>
<tr>
<th>Scheme</th>
<th>Order</th>
<th>( \nu = 0.25 )</th>
<th>( \nu = 0.75 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leapfrog</td>
<td>2nd</td>
<td>25</td>
<td>17</td>
</tr>
<tr>
<td>Upwind Leapfrog</td>
<td>2nd</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>Space Extended</td>
<td>4th</td>
<td>4.7</td>
<td>3.6</td>
</tr>
<tr>
<td>Time Extended</td>
<td>4th</td>
<td>2.1</td>
<td>2.0 (( \nu = 0.4 ))</td>
</tr>
<tr>
<td>Hermitian</td>
<td>4th</td>
<td>2.7</td>
<td>2.1</td>
</tr>
</tbody>
</table>

Table 2.2: Cells-per-Wavelength Requirements for Various Leapfrog Schemes to the Scalar Advection Equation

Figure 2.7: Phase Speed Characteristics for the Space and Time Extended Fourth-Order Upwind Leapfrog Techniques

table, the fourth-order versions all do a much better job than their second-order relatives. In particular, the time-extended scheme shows excellent accuracy - only a little over two cells-per-wavelength required to maintain at most one percent error. Unfortunately it has maximum Courant number limitation of \( \nu = 1/2 \). Jeltsch and Kiani [JK91] have developed a formula for the stability limits of such time extended schemes. Namely the stability is restricted to the Courant number range of \( 0 \leq \nu \leq 1/(k - 2) \) where \( k \) is the number of time levels. Thus this predicts the
three-time level upwind leapfrog scheme as having a stability range of $0 \leq \nu \leq 1$ and the four-time level leapfrog scheme stability range of $0 \leq \nu \leq 1/2$. Likewise a five time-level compact scheme (would be sixth-order accurate) would have a stable Courant number range of $0 \leq \nu \leq 1/3$.

An easier way to visualize this result is based upon a theorem due to Iserles [Ise91] which states that there cannot be a difference of more than one in the number of interpolants on either side of the characteristic through the unknown point in order for a scheme for simple scalar advection to be stable. As the Courant number changes, the location for the interpolants is affected. This leads to there being a difference of two or more interpolants on either side of the characteristic. This effect is illustrated in Figure 2.9. As can be seen, for $\nu > 1/2$ four interpolants end up on the right side of the characteristic path for the update $u_N$ versus one on the left side.

Finally, the Hermitian scheme has some nice properties in that a Courant number up to one may be used, and the compact stencil makes the scheme accurate
and robust at boundaries. Although the Hermitian scheme is more costly as far as memory requirements are concerned since both a point value and a slope update are required, this drawback may be not too much of an issue on parallel machines. Since the spatial extent of the scheme is only one computational cell, one might be able to do a more efficient parallel implementation than would be required for a scheme with a stencil of larger spatial extent.

2.3.2 Fourth-Order Performance Comparisons

For the same model problem as in Section 2.2.3, Figure 2.10 shows results for each of the fourth-order upwind leapfrog schemes. As can be seen, each of the methods does much better than the second-order upwind leapfrog method. The time-extended and Hermitian schemes appear to do a better job than the space extended method. Although some would say that a fourth-order scheme, however
it is achieved, must do much better than any second-order scheme, I believe that keeping the stencils as compact as possible is also important. Although all these schemes are formally fourth-order accurate, factors of two to three improvements in resolution are achievable as can be seen in Table 2.2. This trend can be taken in the same context as the upwind leapfrog scheme being at least two times more accurate than the standard leapfrog. The desire is thus to keep the most compact and upwind-biased arrangement of the data points which comprise the interpolants of the stencils.
CHAPTER III

MORE GENERAL FORMS OF SCALAR ADVECTION

In this chapter, I examine ways in which upwind leapfrog based methods can be developed for more general forms of the scalar advection equation. This includes the effects of source terms and non-constant coefficients. I also examine techniques for developing upwind leapfrog based methods for scalar advection in multiple spatial dimensions.

3.1 Non-constant Advection Speed

I begin by examining how an upwind leapfrog method can be developed for the scalar advection equation with a non-constant advection speed

\[ u_t + a(x)u_x = 0 \]  

(1)

Non-constant coefficients naturally arise in the aeroacoustic equations, but it also turns out that non-constant coefficients appear in the case of non-uniform spacing if one takes the approach of mapping the physical mesh to a uniform computational space mesh as is commonly done for finite-difference methods. Using this approach for simple one-dimensional scalar advection, the equation becomes modified in such
a way that a non-constant metric mesh transformation term ends up multiplying the
$u_x$ gradient term. This is analogous to a non-constant advection speed.

In the case of non-constant coefficients, one can no longer use the dispersion
relation method to ascertain the stability or accuracy of a given method. The mod-
ified equation technique becomes the next best tool to at least check for scheme
consistency and to aid in developing higher-order extensions. In the case of non-
constant coefficients, the modified equation is a bit more difficult to determine as I
demonstrate in the following example.

As in Section 2.1.2, suppose I have an arbitrary second-order scheme that has
the following truncation error which initially has gradient terms involving the time
variable $t$

$$u_t + a(x)u_x = c_{itt}u_{tti} + c_{itx}u_{iti} + c_{lx}u_{txi} + c_{xx}u_{xxi} + O[(\Delta x)^3,(\Delta t)^3]$$

(2)

where $c$ are the coefficients of the third-order gradients of $u$. To first eliminate the
$u_{tti}$ gradient, I can differentiate Equation (2) twice with respect to $t$ which yields

$$u_{tti} + a(x)u_{xiti} = c_{tti}u_{tti} + c_{itx}u_{tti} + c_{lx}u_{txti} + c_{xx}u_{xxti} + O[(\Delta x)^3,(\Delta t)^3]$$

(3)

Substituting this resulting equation for $u_{tti}$ back into Equation (2) leads to a modified
equation having the form

$$u_t + a(x)u_x = d_{tt}u_{tt} + d_{tx}u_{tx} + d_{xx}u_{xx} + O[(\Delta x)^3,(\Delta t)^3]$$

(4)

where $d$ new represent coefficients of the third-order gradient terms for this new
version of the differential equation. To eliminate the $u_{xiti}$ gradient, one might be
inclined to subsequently differentiate Equation (4) with respect to $t$ and $x$ which
would yield

\[ u_{i+1} + a(x)u_{i+1} + a_x(x)u_i = d_{i+1}u_{i+1} + d_{i+1}u_{i+1} + d_{i+1}u_{i+1} + O[(\Delta x)^3, (\Delta t)^3] \]

(5)

However, the second-order gradient term \( u_{i+1} \) arises in this equation. Following the procedure above, one can only eliminate successively higher-order gradient terms. Unfortunately in the case of non-constant coefficients, lower-order gradient terms "crop-up" in the elimination process and thus the procedure cannot be used. This is why I have adopted the matrix method technique as outlined in Chapter II.

One method for discretizing the upwind leapfrog scheme in the case of the non-constant coefficients is to average the wavespeed as \( a \approx (a_{i-1} + a_i)/2 \) which results in the upwind leapfrog scheme having the form

\[
\frac{u_{i+1}^n - u_i^n + u_{i-1}^n - u_{i-1}^{n-1}}{2 \Delta t} + \left( \frac{a_{i-1} + a_i}{2} \right) \frac{u_i^n - u_{i-1}^n}{\Delta x} \]

(6)

If one defines a reference Courant number \( \nu = a_{\infty} \Delta t / \Delta x \), the update \( u_{i+1}^n \) for this scheme can be expressed as

\[
u_{i+1} = u_{i-1} - \nu \left( \frac{a_{i-1} + a_i}{a_{\infty}} \right) \left( u_i^n - u_{i-1}^n \right) \]

(7)

Using the same method of determining the modified equation as outlined in Chapter II, one obtains the following coefficient matrix \( [C] \) (see Section 2.1.2, Equation (25))

\[
\begin{bmatrix}
1 & 0 & -\Delta x & -\Delta x^2 & -\Delta x^2 \\
0 & 1 & 0 & -\Delta x^2 & -\Delta x^2 \\
0 & 0 & 1 & 0 & -\Delta x^2 \\
0 & 0 & 1 & 0 & -\Delta x^2 \\
0 & 0 & 1 & 0 & -\Delta x^2 \\
\end{bmatrix}
\]

(8)

As can be seen, the matrix now has a non-zero element in the lower triangle. Thus the simple substitution process of obtaining the modified equation which works for
constant coefficients can no longer be used. A matrix inversion is necessary. Before I show the resulting modified equation, I would like to take a moment to explain the complexity that arises when trying to determine the equivalent differential equations that are needed in order to develop higher-order finite-difference schemes for equations with non-constant coefficients.

The following table shows how the size of the matrix grows as a function of the number of dimensions and the order of the truncation error desired in addition to whether a scalar or system of equations is being considered. My ultimate objective is to create fourth-order schemes for the three-dimensional aeroacoustic system of equations based upon the upwind leapfrog scheme. As long as the stencils are kept time reversible, then only the gradients associated with terms of second-order must be determined. As will be shown in the following chapter, in three-dimensions, the upwind leapfrog scheme implementation for multi-dimensional acoustics and aeroacoustics requires the storage of six dependent variables - three components of velocity and three pressure variables due to the staggered nature of the mesh to be discussed later. From the table, it can be seen that a $90 \times 90$ symbolic matrix will have to be inverted. Furthermore, in order to “double check” the scheme I proceed to develop is in fact fourth-order, a $210 \times 210$ matrix will have to be inverted. This is to make

<table>
<thead>
<tr>
<th>Order</th>
<th>Number of Dimensions</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>scalar</td>
<td>system</td>
<td>scalar</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>3</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>6</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>10</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>15</td>
<td>30</td>
<td>35</td>
</tr>
</tbody>
</table>

Table 3.1: Sizes of Symbolic Matrices that Must Be Inverted to Determine Modified Equations for Problems with Non-constant Coefficients
sure no third-order terms, which might be the result of a mistake in setting up the fourth-order discretization, appear within the truncation. This unfortunately is beyond the ability of the symbolic mathematical software that is currently available. The current limit to the matrix size that can be inverted using the present software and computers is around $15 \times 15$. I can therefore check the one-dimensional scalar advection scheme that I have created up to fourth-order, and I can develop (but not check) a possible fourth-order scheme for two-dimensional scalar advection. As far as non-constant coefficients, that is about the current limit. For constant coefficients, I can still resort to the substitution technique of determining the equivalent differential equation which I have found to work using the current software and computers at least up to fourth-order in three-dimensions.

For the finite difference scheme (7), inverting (8) and solving for $u_t$ yields the following equivalent differential equation

$$u_t + a u_x = \left[ \frac{\nu (2\nu a - 3a_\infty) (a_{xx} a + a_x^2) u_x}{12a_\infty^2} \right.$$

$$+ \frac{a_x (\nu a - a_\infty) (2\nu a - a_\infty) u_{xx}}{4a_\infty^2} + \frac{a (\nu a - a_\infty) (2\nu a - a_\infty) u_{xxx}}{12a_\infty^2} \right] (\Delta x)^2$$

$$+ \mathcal{O}[(\Delta x)^4]$$

As can be seen, due to the presence of the non-constant wave speed, even ordered (diffusive) gradients of $u$ appear in the truncation error. It is interesting to note however that since the computational stencil remains symmetric, the truncation only has terms involving even powers of $\Delta x$. A fully fourth-order scheme can be created if suitable second-order finite-difference approximations of the gradient terms within truncation error are deducted from the original discretization (7). Before I show how this may be done however, I would like to address the important issue of how
source terms must be treated. In Section 3.3, I will then show how to create and demonstrate a stable fourth-order upwind leapfrog based scheme for both source terms and non-constant coefficients.

3.2 Source Terms

In this section, I look at techniques for using the upwind leapfrog scheme to model scalar advection with a linear source term

\[ u_t + au_x + bu = 0 \]  

(10)

I begin with the case of constant coefficients so that dispersion analysis can be used to study stability and accuracy, and I examine two different approaches for treating the source term. The first and most direct approach is to treat the source term as a simple average of the values at \( u_i \) and \( u_{i-1} \) \( (bu \approx b(u_i + u_{i-1})/2) \) which is simply added to the upwind leapfrog scheme (2.6). The update may be expressed as

\[ u_{i+1}^{n+1} - u_i^{n} = (1 - 2\nu)(u_{i+1}^{n} - u_i^{n}) - b\Delta t(u_{i+1}^{n} + u_i^{n}) \]  

(11)

Since this is a three-level scheme, it has two families of solutions. As usual, one family simulates the differential problem, while the other is spurious. Without any source term, both solutions are neutrally stable. Thus, if there is no strong mechanism for introducing the spurious solution, it will remain small compared with the true solution. With the source term, and the above discretization, it can be shown that the amplification factors \( g_t \) of the true solution, and \( g_s \) of the spurious solution satisfy \( |g_t||g_s| = 1 \). Thus, whenever the true solution should be decreasing \( (b > 0, |g_t| < 1) \), the spurious solution will grow \( (b > 0, |g_s| > 1) \). Even though the spurious solution may only be generated by rounding error, it will in time, overwhelm the true solution.
As was done in Chapter II, if I again consider a periodic solution of the form

\[ u(x, t) = \exp \left[ I \left( \frac{\phi}{\Delta t} - \frac{\theta x}{\Delta x} \right) \right] \]  

(12)

the exact phase speed relation is

\[ \phi = \nu \theta + I b \Delta t \]  

(13)

The dispersion relation for (11) is given by

\[ \sin(\phi - \frac{\theta}{2}) = (1 - 2\nu) \sin \frac{\theta}{2} + I b \Delta t \cos \frac{\theta}{2} \]  

(14)

Explicit representation of \( \phi \) for the discretization is quite complex, and it is much easier to see its structure by plotting the amplification \( g = \exp(I \phi) \) for specific values of \( \nu, \theta, \) and \( b \Delta t \). Figure 3.1a shows the amplification factors for (11) when \( \nu = 1/4, \theta = 2\pi/3 \) \( (N = 3 \ \text{cells-per-wave}) \), and \( b \Delta t = 1/4 \). The square symbol represents the exact amplification, and the circles are the scheme amplification factors. As can be seen, the simple averaging technique for the source term yields a scheme with an amplification which nearly matches the analytical value, and a spurious mode which has growth rate much greater than unity \( |g_s| > 1 \). Thus a mechanism such as round-off error may trigger this spurious mode to grow and overwhelm the true solution. In the following, a sample problem will be presented which demonstrates this behavior.

For simple scalar advection, the problem can be nicely overcome by introducing the transformation

\[ u = e^{-b t} v \quad \Rightarrow \quad v_t + av_x = 0 \]  

(15)

Applying the upwind leapfrog method to update \( v \), and then expressing the result in terms of \( u \), leads to the following algorithm

\[ e^{ib \Delta t} u_{i+1}^{n+1} - e^{-ib \Delta t} u_{i}^{n-1} = (1 - 2\nu)(u_{i+1}^{n} - u_{i}^{n}). \]  

(16)
By construction, I now have $|g_\neq| = |g_l| = e^{-b\Delta t}$, and the spurious solution never grows at the expense of the true one.

A formal generalization of this also is possible when dealing with a system of equations as will be shown Chapter V. In that case, following the procedure above would necessitate the calculation of a matrix exponential which is computationally a very expensive procedure. A much more practical alternative is to abandon the idea that both amplification factors be exact, but retain the condition that they be equal. One way to do this is to replace the factors $e^{\pm b\Delta t}$ by their first-order expansions $1 \pm b\Delta t$. This gives a scheme with

$$|g_\neq| = |g_l| = \sqrt{\frac{1 + b\Delta t}{1 - b\Delta t}} = e^{b\Delta t} + O[(\Delta t)^3]$$  \hspace{1cm} (17)

It is interesting to note that this simple first-order expansion for $e^{\pm b\Delta t}$ is sufficient to make the numerical amplification factors third-order accurate. The resulting scheme

---

**Figure 3.1:** Amplification Factors Plot for the Upwind Leapfrog Scheme to the Scalar Advection Equation with Source Term, $\nu = 1/4$, $\theta = 2\pi/3$ ($N = 3$ cells-per-wave), and $b\Delta t = 1/4$
which I have denoted the “weighted” upwind leapfrog scheme may be expressed as

\[
\frac{(1 + b \Delta t)u_i^{n+1} - u_i^n + u_{i-1}^n - (1 - b \Delta t)u_{i-1}^{n-1}}{2\Delta t} + a \frac{u_i^n - u_{i-1}^n}{\Delta x} = 0
\]  

(18)

or

\[
\frac{(1 + b \Delta t)(u_i^{n+1} - u_i^n) + (1 - b \Delta t)(u_{i-1}^n - u_{i-1}^{n-1})}{2\Delta t} + a \frac{u_i^n - u_{i-1}^n}{\Delta x} + b \frac{u_i^n + u_{i-1}^n}{2} = 0
\]  

(19)

In the first form, the source term does not explicitly appear since it has been absorbed by the transformation approximation. After some arranging, the second form shows how the discretization can be interpreted as a modification to the original upwind leapfrog scheme whereby each of the time-legs of the original leapfrog stencil is weighted by a first-order approximation to \(e^{\pm \Delta t}\).

Figure 3.1b shows the amplification factors for the weighted upwind leapfrog technique again for the conditions \(\nu = 1/4, \theta = 2\pi/3\) (\(N = 3\) cells-per-wave), and \(b\Delta t = 1/4\). As can be clearly seen, the spurious mode no longer has an amplification greater than one. Also the true mode matches the exact amplification much better than the simple averaging source term method.

### 3.2.1 Spherical Wave Equation

The next issue is whether this local analysis provides a valid scheme for problems with non-constant coefficients. To test this I have calculated solutions to the equation

\[
u_t + u_r + \frac{u}{r} = 0,
\]  

(20)

which represents spherically symmetric outgoing waves. Its general solution is

\[u(r,t) = \frac{f(r-t)}{r}.
\]  

(21)

Figure 3.2 shows results for simulations of this equation using the simple averaging of source term upwind leapfrog method (11) and the weighted time-legs version (19).
I have imposed sinusoidal boundary conditions at $r = 5$ and computed the solution in a domain $5 < r < 300$. The source term coefficient $1/r$ for each of the methods is taken as the value midway between the mesh points $r_i$ and $r_{i-1} (1/r_{i-1/2})$

With the simple averaging version of the scheme, an instability sets in after a few hundred time-steps as can be seen from Figure 3.2, and although its amplitude never becomes catastrophic, the solution is very poor for the region $r > 100$. With the weighted legs version, I can run the code on a larger domain, and the solution reaches an asymptotic state after approximately 500 time-steps. I have tested the scheme another 13,000 iterations and the solution envelope shows no further changes.

![Figure 3.2: Upwind Leapfrog Techniques for the Spherical Wave Equation ($\Delta x = 1$, $t = 2000$, $\nu = \Delta t/\Delta r = 3/4$)](image)

### 3.2.2 Higher-Order Accuracy

I now examine how fourth-order accuracy may be achieved in the case of a source term. I again take the approach of using the modified equation. The modified
equation for the unstable non-weighted upwind leapfrog stencil is

\[ u_t + au_x + bu = \left[ \frac{b^3\nu^2}{6a^2}u + \frac{b^2\nu(2\nu - 1)}{4a}u_x + \frac{b\nu(\nu - 1)}{2}u_{xx} \right. \]

\[ + \frac{(2\nu - 1)(\nu - 1)}{12}u_{xxx} \left. \right] (\Delta x)^2 \]

\[ - \left[ \frac{3b^5\nu^4}{40a^4}u + \frac{3b^4\nu^3(2\nu - 1)}{16a^3}u_x + \frac{b^3\nu^2(6\nu^2 - 6\nu + 1)}{8a^2}u_{xx} \right. \]

\[ + \frac{b^2\nu(2\nu^2 - 1)(18\nu^2 - 18\nu + 1)}{48a}u_{xxx} \]

\[ + \frac{b\nu(\nu - 1)(3\nu - 1)(3\nu - 2)}{24a}u_{xxxx} \left. \right] (\Delta x)^4 \]

\[ + \mathcal{O}[(\Delta x)^6] \]  

and for the weighted time legs approach

\[ u_t + au_x + bu = \left[ \frac{b^3\nu^2}{3a^2}u - \frac{b^2\nu(2\nu - 1)}{4a}u_x + \frac{(2\nu - 1)(\nu - 1)}{12}u_{xx} \right. \]

\[ \left. \frac{(2\nu - 1)(\nu - 1)}{12}u_{xxx} \right] (\Delta x)^2 \]

\[ - \left[ \frac{b^5\nu^4}{5a^4}u + \frac{3b^4\nu^3(2\nu - 1)}{16a^3}u_x - \frac{b^3\nu^2(6\nu^2 - 6\nu + 1)}{48a}u_{xx} \right. \]

\[ + \frac{a(\nu - 1)(2\nu - 1)(3\nu - 1)(3\nu - 2)}{240}u_{xxxx} \left. \right] (\Delta x)^4 \]

\[ + \mathcal{O}[(\Delta x)^6] \]  

I present both equivalent differential equations up to fourth-order in order to show some interesting features of the weighted approach. One prominent feature to be noted is that the weighted technique has no even-ordered diffusive gradient terms within its truncation. In fact, I have calculated its modified equation up to tenth-order, and no even-order gradients of \( u \) appear in the truncation. This is not to say they should not appear in the truncation, it is just an interesting feature when using the simple weightings. Also, even though a first-order expansion for \( e^{\pm i\Delta t} \) is used for the weightings, the scheme has no odd-ordered terms of \( \Delta x \) in its truncation. This means if I eliminate the terms of \( \mathcal{O}[(\Delta x)^2] \) in the truncation error via suitable finite-
difference representations of the gradient terms multiplied by \((\Delta x)^2\), the scheme will in fact be fourth as opposed to third-order. This is interesting in that one would expect the need to use a higher-order expansion of \(e^{\pm i\Delta t}\) for the weightings when a higher-order method is being sought. If I use the exponential \(e^{\pm i\Delta t}\) for the weightings in the discretization, the equivalent differential equation is

\[
\begin{align*}
    u_t + au_x + bu &= \left[ \frac{b^3 \nu^2}{6a^2}u - \frac{b^3 \nu}{4a}u_x + \frac{(2\nu - 1)(\nu - 1)}{12}u_{xxx} \right](\Delta x)^2 \\
    &+ \left[ \frac{b^5 \nu^4}{120a^4}u + \frac{b^4 \nu^3}{48a^3}u_x + \frac{b^3 \nu^3(\nu - 1)}{12a^2}u_{xx} \\
    &+ \frac{b^2 \nu(2\nu^2 - 1)(6\nu^2 - 6\nu + 1)}{48a}u_{xxx} \\
    &+ \frac{a(\nu - 1)(2\nu - 1)(3\nu - 1)(3\nu - 2)}{240}u_{xxxxx} \right](\Delta x)^4 \\
    &+ \mathcal{O}[(\Delta x)^6]
\end{align*}
\]

(24)

When using the exact weighting, even-ordered diffusive gradient terms do appear in the truncation. Thus I find the property that no even-ordered gradient terms appear in (23) all the more interesting.

### 3.3 Non-constant Coefficients and Source Terms

In this section, I construct a fully fourth-order upwind leapfrog scheme for scalar advection with both non-constant coefficients and a source term.

\[
    u_t + a(x)u_x + b(x)u = 0
\]

(25)

There are a multitude of ways in which to discretize the non-constant coefficients. For example, in using the upwind leapfrog scheme, I could choose the wavespeed \(a\) at \(x_i\) or \(x_{i-1}\) which I will denote as \(a_i\) or \(a_{i-1}\). But clearly an average of the two \((a_i + a_{i-1})/2\) is what is required to achieve second-order accuracy. As for the source term, I treat it in the same manner as was described in the last section by weighting
each of the time legs. The weightings are taken as $1 \pm (b_i + b_{i-1}) \Delta t/2$. The overall scheme may be written as

$$
\frac{(1 + \frac{b_i + b_{i-1}}{2} \Delta t)(u_{i+1}^{n+1} - u_i^n) + (1 + \frac{b_i + b_{i-1}}{2} \Delta t)(u_{i-1}^n - u_i^{n-1})}{2 \Delta t}
$$

$$
\frac{1}{2 \Delta x} (a_{i-1} + a; (u_i^n - u_{i-1}^n) + \frac{1}{4} (b_{i-1} + b; (u_i^n + u_{i-1}^n))
$$

This defines the basic second-order scheme.

To create a higher-order method, I first determine the modified equation which turns out to be

$$
u_t + au_x + bu = \frac{(\Delta x)^2}{12 a_x^2}(f_1 u + f_2 u_x + f_3 u_{xx} + f_4 u_{xxx}) + O[(\Delta x)^4]
$$

(27)

where

$$f_1 = 2\nu^3 (a^2 b_{xx} + a a_x b_x - 2b^3) - 3a \nu^2 (a_x b_x + a b_{xx})$$

$$f_2 = 3b_x a^2 - 2\nu^2 (3ab^3 - aa_x^2 - a^2 a_{xx} - 3a^2 b_x)$$

$$- 3 a \nu (3ab_x + aa_{xx} + a_x^2 - b^2)$$

$$f_3 = 3a_x (\nu a - a_\infty)(2\nu a - a_\infty)$$

$$f_4 = a (\nu a - a_\infty)(2\nu a - a_\infty)$$

(28)

Here again $a_\infty$ is a reference wave speed used to define the Courant number $\nu = a_\infty \Delta t/\Delta x$. There are no third-order error terms, so a fourth-order scheme can be obtained by subtracting a numerical estimate of the leading error. Since each of the terms that are multiplied by $(\Delta x)^2$ in the truncation error are known, I can eliminate them from the second-order discretization by first making suitable finite-difference representations of them, and then subtracting these representations multiplied by $(\Delta x)^2$ from the difference scheme. For example, if I approximate the wave speed
coefficient $a$ along with its derivatives as

\begin{align*}
a &= \frac{a_i + a_{i-1}}{2} + \mathcal{O}[(\Delta x)^2] \\
a_x &= \frac{a_i - a_{i-1}}{\Delta x} + \mathcal{O}[(\Delta x)^2] \\
a_{xx} &= \frac{a_{i+1} - a_i - a_{i-1} + a_{i-2}}{\Delta x^2} + \mathcal{O}[(\Delta x)^2] \\
a_{xxx} &= \frac{a_{i+1} - 3a_i + 3a_{i-1} - a_{i-2}}{\Delta x^3} + \mathcal{O}[(\Delta x)^2]
\end{align*}

(29)

with similar expressions for the derivatives of $b$ and $u$. Because the errors already contain a factor $(\Delta x)^2$ the estimates need only be, as above, second-order.

I demonstrate the resulting scheme on the problem

\begin{equation}
\frac{u_t}{2 + \cos(2\pi x)} + \sin(2\pi x)u = 0.
\end{equation}

(30)

with initial condition

\begin{equation}
u(x, 0) = \sin(2\pi x)
\end{equation}

(31)

This equation unfortunately does not have an explicit solution; however, a numerical solution that repeats with period 2.0 can be determined using a root finding technique. By using this with an extremely small tolerance, I can essentially get an exact solution. Figure 3.3 shows the exact solution at steps of a quarter of a cycle over the course of one period. As can be seen, the change in wavespeed and the effect of the source term cause the sine wave to become quite modified in shape over the course of a cycle.

Figure 3.4 shows the results of various finite-difference schemes applied to the model problem at $t = 99$, halfway through the fiftieth cycle. I choose this time since the objective of our schemes is to be able to move data on the order of 10 to 100 wavelengths, and furthermore, any errors look most noticeable at mid-cycle. In each
case, a mesh of 50 cells and a Courant number of $\nu = a_\infty \Delta t / \Delta x = 1/4$ has been used. All of the three time-level schemes use the two time-level Lax-Wendroff scheme for their initial steps. In the figure, the solid and dashed lines represent the exact solution and numerical solutions respectively.

Figure 3.4a shows the rather poor results for the Lax-Wendroff scheme. In this version of the Lax-Wendroff scheme, the source term was simply added as $b_i u_i^n$ to the standard Lax-Wendroff scheme for scalar advection. The results show that the scheme appears to be going unstable. Better results may be obtained by a better treatment of the source term. Next, Figure 3.4b shows the results of the standard leapfrog scheme. Again, for this tested form of the standard leapfrog scheme, the source term was explicitly added as $b_i u_i^n$. Although the solution appears to be stable, there is a large quantity of high frequency noise. The solution was evolved for several hundred more wavelengths, yet never went unstable.

Figures 3.4c,d show results for upwind leapfrog scheme and the weighted time leg version of the upwind leapfrog scheme. Although there appear to be no differences in the solutions, a close examination of the errors proves otherwise. The weighted time leg version produces slightly better results. Both schemes are stable, yet it can be seen that the dispersion error has become quite significant.
Finally, Figures 3.4e,f show the results for the fourth-order space-extended upwind leapfrog schemes. In Figure 3.4e, the space-extended fourth-order version of the upwind leapfrog scheme (Equation (61)) has been used with the wave speed simply treated as $a = (a_i + a_{i-1})/2$ and source term added as $bu = [(b_i + b_{i-1})(u^n_i + u^n_{i-1})]/4$. As can be seen, this is an improvement over the upwind leapfrog and weighted upwind leapfrog schemes; however, this scheme does not account for the terms arising from variations in $a, b$ and is still formally only second-order. Finally, Figure 3.4f shows how the formally fourth-order technique developed above, handles this problem very well.

Figure 3.4: Various Finite-Difference Techniques for Model Problem ($t = 99$, $a_\infty = 1/2$, $\nu = a_\infty \Delta t/\Delta x = 1/4$)
3.4 Multiple Dimensions

The remainder of this chapter is devoted to multi-dimensional aspects for scalar advection. In this section, I look at the feasibility of using the upwind leapfrog scheme to model the two-dimensional scalar advection equation

\[ u_t + a(x, y)u_x + b(x, y)u_y = 0 \]  

(32)

To begin with, I will look at the case where \( a \) and \( b \) are constants, such that the advection direction is given by

\[ \alpha = \tan^{-1} \left( \frac{a}{b} \right) \]  

(33)

3.4.1 Mesh and Stencil Strategies

There are a number of different strategies for creating upwind leapfrog based schemes for multi-dimensional advection which primarily depend on the type of computational mesh one employs. Figure 3.5 illustrates two different mesh arrangements one may use. The first mesh is a typical uniform arrangement where the dependent variables are stored at the computational cell corners. The second mesh type is known as a staggered arrangement in which the dependent variables are stored at each of the cell faces. For each of the mesh strategies, the total number of computational mesh points \( N_T \) over a uniform mesh that spans \( N \) computational cells in each coordinate direction is given by

\[ N_{T \text{non-staggered}} = (N + 1)^d \quad \text{and} \quad N_{T \text{staggered}} = d(N)^{d-1}(N + 1) \]  

(34)

where \( d \) is the number of spatial dimensions. As can be seen, as \( N \) gets large, the staggered mesh requires roughly \( d \) times the number of mesh points as the non-staggered approach. I have found however that the staggered approach is usually
Figure 3.5: The Upwind Leapfrog Scheme for Two Dimensional Advection

around \(d\) times more accurate than the non-staggered arrangement, and this makes up for the additional memory cost. Also the staggered arrangement turns out to be the preferred method for implementing upwind leapfrog based methods for stationary and non-stationary acoustics as will be shown in the following chapter.

Figure 3.6 illustrates the computational stencils for an upwind leapfrog approach on non-staggered meshes. In this case, each of the computational stencils is aligned with the coordinate axes. Depending upon the advection direction, I might choose one or a combination of the four upwind leapfrog stencils. If the advection is in the direction of the positive \(x\) axis, the best choice for the update would come solely from the stencil labeled “W” (for West) in the figure. For this particular stencil, the advection equation is discretized as

\[
\begin{align*}
\frac{u_{i,j}^{n+1} - u_{i,j}^n + u_{i-1,j}^n - u_{i-1,j-1}^n}{2\Delta t} + \frac{u_{i,j}^n + u_{i-1,j}^n}{\Delta x} & + \frac{u_{i,j+1}^n - u_{i,j-1}^n + u_{i-1,j+1}^n - u_{i-1,j-1}^n}{4\Delta y} = 0
\end{align*}
\]

(35)
If the advection direction varies, I have found it best to use a blending of the stencils to avoid the abrupt changes in discretization of the update. Blending is also absolutely necessary for the diagonal aligned and staggered mesh upwind leapfrog based form that I show next since there is no “overlapping” of the directional stencils. A simple and effective trigonometric blending can be formulated as follows.

If I let the capital letters $N$, $S$, $E$, and $W$ denote the directions North, South, East, and West respectively, $w$ represent blending weight, and $s$ denote scheme or stencil, then the net update may be expressed as the sum of updates from each of the stencils

$$
\text{net scheme} = w_N s_N + w_S s_S + w_E s_E + w_W s_W \tag{36}
$$

An easy way to set the blendings is to have

$$
w_S = \sin^2(\alpha), \quad w_N = 0 \quad 0 \leq \alpha \leq \pi
$$

$$
w_S = 0, \quad w_N = \sin^2(\alpha) \quad -\pi \leq \alpha \leq 0
$$

$$
w_W = \cos^2(\alpha), \quad w_E = 0 \quad -\pi/2 \leq \alpha \leq \pi/2
$$

$$
w_W = 0, \quad w_E = \cos^2(\alpha) \quad -\pi \leq \alpha \leq -\pi/2 \quad \text{or} \quad \pi/2 \leq \alpha \leq \pi \tag{37}
$$
Since the overall stencil is no longer symmetric in space and time when blending is used, the resulting scheme becomes slightly dissipative.

Figure 3.7 illustrates the computational stencils for an upwind leapfrog approach that can also be used on non-staggered meshes. In this instance, each of the computational stencils is aligned in a direction oblique to the mesh. This form has an elegant feature in that it may be used at computational boundaries. The best update if the advection direction is at an angle $45^\circ$ to the mesh would come solely from the stencil labeled SW in the figure. Its discretization may be expressed as

$$
\frac{u_{i,j}^{n+1} - u_{i,j}^n + u_{i-1,j-1}^n - u_{i-1,j-1}^{n-1}}{2\Delta t} + a \frac{u_{i,j}^n + u_{i-1,j}^n + u_{i-1,j-1}^n + u_{i-1,j-1}^{n-1}}{2\Delta x} \\
+ b \frac{u_{i,j}^n - u_{i,j-1}^n + u_{i-1,j}^n - u_{i-1,j-1}^n}{2\Delta y} = 0
$$

In this case, the trigonometric blending of the stencils should proceed as

$$
w_{SW} = \sin^2(\alpha + \pi/4), \ w_{NE} = 0 \quad -\pi/4 \leq \alpha \leq 3\pi/4
$$
$$
w_{SW} = 0, \ w_{NE} = \sin^2(\alpha + \pi/4) \quad -\pi \leq \alpha \leq -\pi/4 \ or \ 3\pi/4 \leq \alpha \leq \pi
$$
$$
w_{NW} = \cos^2(\alpha + \pi/4), \ w_{SE} = 0 \quad -3\pi/4 \leq \alpha \leq \pi/4
$$
$$
w_{NW} = 0, \ w_{SE} = \cos^2(\alpha + \pi/4) \quad \pi/4 \leq \alpha \leq \pi \ or \ -\pi \leq \alpha \leq -3\pi/4
$$
The staggered mesh upwind leapfrog scheme is implemented with computational stencils as shown in Figure 3.8. As can be seen, the staggering also enables the scheme to have a very compact stencil. If \((x(i, j), y(i, j))\) corresponds to the center of computational cell \(i, j\) the discretization for the stencil to be used when the advection direction is in the direction of the positive \(x\) axis is given by

\[
\frac{u_{i+1/2,j}^{n+1} - u_{i+1/2,j}^n + u_{i-1/2,j}^n - u_{i-1/2,j}^{n-1}}{2\Delta t} + a \frac{u_{i+1/2,j}^n + u_{i-1/2,j}^n}{\Delta x} + b \frac{u_{i,j+1/2}^n - u_{i,j-1/2}^n}{\Delta y} = 0
\]

(40)

Since the stencils are aligned with the coordinate axes, the blending form as given in Equation (37) should be used.

### 3.4.2 Dispersion Analysis

For two-dimensional dispersion analysis of the scalar advection equation, the solution is assumed to be composed of periodic data of the form

\[
u(x, y, t) = \exp \left[ I \left( \frac{\phi}{\Delta t} - \frac{\theta_x x}{\Delta x} - \frac{\theta_y y}{\Delta y} \right) \right]
\]

(41)

The exact phase speed \(\phi_c\) for the constant coefficient form of Equation (32) is

\[
\phi_c = \nu_x \theta_x + \nu_y \theta_y
\]

(42)
where \( \nu_x = a \Delta t / \Delta x \) and \( \nu_y = b \Delta t / \Delta y \) represent the coordinate aligned Courant numbers.

In the figures that follow, I consider the dissipation and phase speed properties of each of the schemes as the advection direction \( \alpha = \tan^{-1} \left( \frac{x}{y} \right) \) and wave front direction coincide and are varied between \(-90^\circ\) and \(90^\circ\). The coordinate Courant numbers are a bit difficult to define in such a case since the advection directions are continually changing. I however choose them as

\[

\nu_x = \frac{1}{4} \cos(\alpha) \quad \text{or} \quad \nu_y = \frac{1}{4} \sin(\alpha)

\]

which corresponds to Courant number of \(1/4\) when the advection directions are aligned with the coordinate axes. The Fourier angles \( \theta_x \) and \( \theta_y \) are related to a “base” cells-per-wavelength \( N \) and the wavefront direction \( \alpha \) via the following relations

\[

\theta_x = \frac{2\pi}{N} \cos(\alpha) \quad \text{or} \quad N_x = \frac{N}{\cos(\alpha)}

\]

\[

\theta_y = \frac{2\pi}{N} \sin(\alpha) \quad \text{or} \quad N_y = \frac{N}{\sin(\alpha)}

\]

I have chosen 4, 8, and 16 for the values of \( N \) to give an indication of order of accuracy trends. The dissipation error \( D \) is a bit more difficult to work with in multi-dimensions, and I instead consider an overall dissipation error which I define as

\[

D\% = \left[ 1 - \left( e^{-\phi_I} \right)^{\frac{N}{\nu}} \right] \times 100

\]

where \( \phi_I \) is the imaginary component of the numerical phase relation. The phase error I again define as

\[

E\% = \left( \frac{\phi_R}{\nu_x \theta_x + \nu_y \theta_y} - 1 \right) \times 100

\]
where $\phi_R$ is the imaginary component of the numerical phase relation.

Figures 3.9 (non-blended) and 3.10 (blended) show the error trends for the coordinate aligned non-staggered mesh upwind leapfrog methods. As can be seen, when there is no blending, the scheme is neutrally stable. When blending is employed however, there turns out to be a bit of dissipation. It should be noted however that the magnitude of the dissipation error is extremely small in comparison to the phase speed errors. $D$ and $E$ differ by approximately two orders of magnitude as can be seen from Figure 3.10. The blending can also be seen to help keep the phase error trends smooth as the advection direction varies.

Figure 3.9: Error Trends for the Non-Blended Coordinate Aligned Non-staggered Mesh Upwind Leapfrog Methods

The diagonally aligned blended non-staggered mesh upwind leapfrog method has dissipation and phase speed properties as shown in Figure 3.11. Although the diagonally aligned form has the nice property of a very compact stencil which may be used at computational boundaries, it unfortunately has about twice the error for both dissipation and phase as the blended coordinate aligned method. A scheme
that blends both the coordinate and diagonal aligned stencil might prove to be a good compromise.
The weighted staggered mesh upwind leapfrog scheme has error trends as shown in Figure 3.12. When compared to either of the un-staggered mesh methods, it can be seen to have easily twice the resolution which helps to make up for the added cost of the staggered mesh. It is also interesting to compare these schemes to the standard leapfrog scheme whose dissipation and phase error trends are shown in Figure 3.13. As can be seen, the standard leapfrog scheme has no dissipation error, but its phase speed properties are very poor when compared to any of the upwind leapfrog methods.

3.4.3 Non-Constant Coefficients and Fourth-Order Accuracy

I now look into developing some higher-order variations of the two-dimensional upwind leapfrog based methods for scalar advection. To make the methods more general, I also consider non-constant coefficients. To develop the higher-order methods, I again take the approach of determining the equivalent differential equation for a “base” second-order scheme and subsequently deduct from the scheme suitable finite
Figure 3.13: Amplitude and Phase Error Trends for the Standard Leapfrog Scheme

difference representations of the terms in the truncation error I wish to eliminate.

For example, if I begin with the W stencil for the coordinate aligned non-staggered mesh upwind leapfrog stencil and average the advection speeds $a$ and $b$ such that the scheme is

\[
\frac{u_{i,j}^{n+1} - u_{i,j}^n + u_{i-1,j}^n - u_{i-1,j}^{n-1}}{2 \Delta t} + \frac{1}{2 \Delta x}(a_{i,j} + a_{i,j-1})(u_{i,j}^n + u_{i-1,j}) + \frac{1}{8 \Delta y}(b_{i,j} + b_{i,j-1})(u_{i,j+1}^n - u_{i,j-1}^n + u_{i-1,j+1} - u_{i-1,j-1}) = 0
\]

the equivalent differential equation for this discretization over a uniform mesh ($\Delta x =$
As can be seen, the truncation error for the two-dimensional schemes is quite complex. However by expanding the computational stencil as shown in Figure 3.14, it is possible to come up with suitable gradient approximations to eliminate all the terms of $O(h^2)$ within the truncation error to create a fourth-order scheme. Blending of the stencils is still necessary which introduces a bit of dissipation although very small in magnitude. With similar knowledge of the modified equations for the diagonally aligned non-staggered and staggered upwind leapfrog schemes, one can also create fourth-order methods based on the stencils shown in Figures 3.15 and 3.16.

To get an idea of the performance of these higher-order methods, Figures 3.17 and 3.18 show error trends for the coordinate aligned non-staggered (Figure 3.14) and staggered methods (Figure 3.16). Again I consider the dissipation error $D$ and...
Figure 3.14: A Fourth-Order Coordinate Axis Aligned Non-Staggered Upwind Leapfrog Stencil

Figure 3.15: A Fourth-Order Mesh Oblique Non-Staggered Upwind Leapfrog Stencil

Figure 3.16: A Fourth-Order Staggered Upwind Leapfrog Stencil
phase speed $E$ as the advection direction and wave front angles coincide and vary from $-90^\circ$ and $90^\circ$ for coordinate aligned number of cells-per-wavelength $N = 4$, 8, and 16.

\[(E\%) \times 10^2\]

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure317}
\caption{(a) Dissipation Error (b) Phase Speed Error}
\end{figure}

Figure 3.17: Error Trends for the Fourth-Order Blended Coordinate Aligned Non-staggered Mesh Upwind Leapfrog Methods

Compared to any of the second-order methods (Figures 3.9-3.13), these fourth-order variations can be seen to perform much better. This is especially true for the staggered mesh arrangement where four cells-per-wavelength is all that is required to keep the phase error to within two percent for any advection direction.

### 3.4.4 Solid Body Rotation Problem

A nice problem for testing the ability of each of the schemes to handle non-constant coefficient multi-dimensional scalar advection is the following equation which represents solid-body rotation

$$u_t + y u_x - x u_y = 0$$

(50)
This equation advects an initial distribution along circles centered at the origin of the coordinates. I have chosen the following initial condition which represents a Gaussian hill centered at \((x = 1/2, y = 0)\).

\[
u(x, y, t = 0) = \exp \left\{-100 \left[ \left( x - \frac{1}{2} \right)^2 + y^2 \right] \right\}
\]

(51)

In the following simulations (Figures 3.19 - 3.21), I have considered the advection of the initial condition over one rotation. I have chosen a uniform mesh consisting of 40 computational cells in each coordinate direction, and I have taken a time-step of \(\pi/10\). 400 time steps thus correspond to one period of rotation.

Figures 3.19a and b show the results for the standard leapfrog and the coordinate aligned non-staggered upwind leapfrog (Figure 3.6) methods. The standard leapfrog scheme has a great deal of difficulty handling this problem. The coordinate aligned non-staggered upwind leapfrog can be seen to do significantly better. Figures 3.20a and b show the results for the diagonally aligned non-staggered mesh and staggered
mesh second-order upwind leapfrog forms. Again, compared to the standard leapfrog method, any of the upwind leapfrog techniques can be seen to perform much better.

Figures 3.21 a and b show the results for the fourth-order non-staggered mesh and staggered mesh schemes. As expected from the dispersion analysis of the constant coefficient form, the fourth-order staggered mesh technique does the best job in preserving the Gaussian hill.

3.4.5 Multiple Dimensions with a Source Term

In the final section of this chapter, I examine how to treat a source term in the discretization of the upwind leapfrog methods for the case of constant coefficient multi-dimensional advection

\[ u_t + au_x + bu_y + cu = 0 \]  

(52)

It turns out, just like for the one-dimensional formulation, the time legs of the computational stencils must be weighted in an appropriate fashion in order to ensure stability. I demonstrate the technique for the second-order scheme as implemented on a staggered mesh. The technique is identical for the non-staggered mesh forms as well as the higher-order variations. Without weighting the time legs, the scheme for the computational stencil aligned with the positive \( x \) can be written as

\[
\frac{(u_{i+1/2,j}^{n+1} - u_{i+1/2,j}^n) + (u_{i-1/2,j}^n - u_{i-1/2,j}^{n-1})}{2\Delta t}
\]

\[
a \frac{u_{i+1/2,j}^n + u_{i-1/2,j}^n}{\Delta x} + b \frac{u_{i,j+1/2}^n - u_{i,j-1/2}^n}{\Delta y} = 0
\]

\[
\frac{c}{4}(u_{i+1/2,j}^n + u_{i-1/2,j}^n + u_{i,j+1/2}^n + u_{i,j-1/2}^n)
\]

(53)

When the time legs are weighted by \( 1 \pm c\Delta t \) as in the one-dimensional case, the scheme can be written as

\[
\frac{(1 + c\Delta t)(u_{i+1/2,j}^{n+1} - u_{i+1/2,j}^n) + (1 - c\Delta t)(u_{i-1/2,j}^n - u_{i-1/2,j}^{n-1})}{2\Delta t}
\]
\[
\begin{align*}
+ a \frac{u_{i+1/2,j}^n + u_{i-1/2,j}^n}{\Delta x} + b \frac{u_{i,j+1/2}^n - u_{i,j-1/2}^n}{\Delta y}
+ \frac{c}{4} (u_{i+1/2,j}^n + u_{i-1/2,j}^n + u_{i,j+1/2}^n + u_{i,j-1/2}^n)
\end{align*}
\]

Again, one may use dispersion relation analysis to determine the stability and accuracy of each of these discretizations. For the case of a constant coefficient source term, the exact dispersion relation is given by

\[
\phi_\epsilon = \nu_x \theta_x + \nu_y \theta_y + I c \Delta t
\]  

(55)

As can be seen from Figure 3.22a, the non-weighted technique has a spurious mode with an amplification greater than unity. Again, round-off error or a boundary condition may trigger this mode which would eventually overwhelm the true solution. Figure 3.22b shows how weighting the time legs can once again stabilize the basic method.
Figure 3.19: Solid Body Rotation Advection Problem (40 × 40 Mesh)
Figure 3.20: Solid Body Rotation Advection Problem Using Upwind Leapfrog Based Stencils ($40 \times 40$ Mesh)
Figure 3.21: Solid Body Rotation Advection Problem using Fourth-Order Upwind Leapfrog Techniques ($40 \times 40$ Cells)
Figure 3.22: Amplification Factors Plot for the Upwind Leapfrog Scheme to the Two-Dimensional Scalar Advection Equation with a Constant Coefficient Source Term (\(c\Delta t = 1/3, \nu_x = 1/4, \nu_y = 3/8, \theta_x = 2\pi/5 \ (N_x = 5), \) and \(\theta_y = 3\pi/8 \ (N_y = 16/3)\)
CHAPTER IV
ACOUSTICS IN STATIONARY AND NON-STATIONARY MEDIA

4.1 Method Development for Acoustics in a Stationary Medium

As mentioned in the introduction, stationary acoustics can be modeled by a first-order system of equations with constant coefficients (Equations (1.12) and (1.13)). In two-dimensions, this leads to a system of three equations for the pressure $p$ and the two components of velocity $u$ and $v$ having the form (dropping the tilde over the transient variables)

\[
\frac{\partial p}{\partial t} = -\rho_0 a_0^2 \left( \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) \quad (1a)
\]

\[
\frac{\partial u}{\partial t} = -\frac{1}{\rho_0} \frac{\partial p}{\partial x} \quad (1b)
\]

\[
\frac{\partial v}{\partial t} = -\frac{1}{\rho_0} \frac{\partial p}{\partial y} \quad (1c)
\]

The first issue to address is how to cast this system into a form that can be readily discretized using concepts of the upwind leapfrog scheme. In its present form, the system lends itself to be more readily discretized using central schemes such as the standard leapfrog scheme

\[
\frac{p_{i,j}^{n+1} - p_{i,j}^{n-1}}{2\Delta t} = -\rho_0 a_0^2 \left( \frac{u_{i+1,j}^n - u_{i-1,j}^n}{2\Delta x} + \frac{v_{i,j+1}^n - v_{i,j-1}^n}{2\Delta y} \right) \quad (2a)
\]
or Yee’s scheme [Yee66] which is very similar to the standard leapfrog scheme yet is constructed over a staggered mesh in space and time

\[
\frac{u^{n+1}_{i,j} - u^{n}_{i,j}}{2\Delta t} = -\frac{1}{\rho_0} \left( \frac{p^n_{i+1,j} - p^n_{i-1,j}}{2\Delta x} \right) \tag{2b}
\]

\[
\frac{v^{n+1}_{i,j} - v^{n}_{i,j}}{2\Delta t} = -\frac{1}{\rho_0} \left( \frac{p^n_{i+1,j+1} - p^n_{i+1,j-1}}{2\Delta y} \right) \tag{2c}
\]

Yee’s technique is very efficient in that each dependent variable is not stored at every mesh point, and the staggering of the mesh enables the scheme to have the same accuracy of the standard leapfrog method when using one half the mesh density.

### 4.1.1 Bicharacteristic Approach

In order to use the upwind leapfrog method for the acoustic system it is necessary to cast the system into a more characteristic like form resembling the scalar advection equation. This may be done using the concept of bicharacteristics. Courant and Hilbert [CH62] provide an excellent discussion of the theory of bicharacteristics. Butler [But60], Chuskin [Chu68], Johnston and Pal [JP72], and Camarero [Cam73] have used bicharacteristics to create numerical methods in the spirit of characteristic methods. Recent contributions have also been made by Dadone and Moretti [DM88] and Parpia, Kentzer and Williams [PKW88]. A bicharacteristic equation can be formed...
by multiplying Equations (1a) - (1c) by respectively \( \rho_0 a_0 \cos \beta \), and \( \rho_0 a_0 \sin \beta \), and subsequently adding them together

\[
\left( \frac{\partial}{\partial t} + a_0 \cos \beta \frac{\partial}{\partial x} + a_0 \sin \beta \frac{\partial}{\partial y} \right) \left[ p + \rho_0 a_0 (u \cos \beta + v \sin \beta) \right] = -\rho_0 a_0^2 \left[ (\sin \beta \frac{\partial}{\partial x} - \cos \beta \frac{\partial}{\partial y})(u \sin \beta - v \cos \beta) \right]
\]

(4)

For any direction \( \beta \), the left hand side resembles a one-dimensional characteristic equation, written along a generator of the Mach cone

\[
dx^2 + dy^2 = a_0^2 dt^2
\]

(5)

and the right hand side expresses derivatives tangential to that cone. In two-dimensions, there are four cases where the generator is contained in a coordinate plane, and this results in four respective bicharacteristic equations aligned with each of the coordinate axes

\[
\frac{\partial(p + \rho_0 a_0 u)}{\partial t} + a_0 \frac{\partial(p + \rho_0 a_0 u)}{\partial x} = -\rho_0 a_0^2 \frac{\partial v}{\partial y}
\]

(6a)

\[
\frac{\partial(p - \rho_0 a_0 u)}{\partial t} - a_0 \frac{\partial(p - \rho_0 a_0 u)}{\partial x} = -\rho_0 a_0^2 \frac{\partial v}{\partial y}
\]

(6b)

\[
\frac{\partial(p + \rho_0 a_0 v)}{\partial t} + a_0 \frac{\partial(p + \rho_0 a_0 v)}{\partial y} = -\rho_0 a_0^2 \frac{\partial u}{\partial x}
\]

(6c)

\[
\frac{\partial(p - \rho_0 a_0 v)}{\partial t} - a_0 \frac{\partial(p - \rho_0 a_0 v)}{\partial y} = -\rho_0 a_0^2 \frac{\partial u}{\partial x}
\]

(6d)

To each of these bicharacteristic forms, one may then apply a characteristic method like the upwind leapfrog scheme to the left hand side. One problem however is readily apparent, and is a typical difficulty in the application of bicharacteristic methods. Four equations emerge that are convenient to work with, but only three are needed