An Adaptive Multilevel Method for Boundary Layer Meteorology

By

David Eric Stevens

A dissertation submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

University of Washington

1994

Approved by _____
(Chairperson of Supervisory Committee)

_____

_____

Program Authorized
to Offer Degree Department of Applied Mathematics

Date 12/14/94

In presenting this dissertation in partial fulfillment of the requirements for the doctoral degree at the University of Washington, I further agree that extensive copying of this dissertation is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for copying or reproduction of this dissertation may be referred to University Microfilms, 1490 Eisenhower Place, P.O. Box 975, Ann Arbor, MI 48106, to whom the author has granted "the right to reproduce and sell (a) copies of the manuscript in microform and/or (b) printed copies of the manuscript made from microform."

Signature David Stern

Date 12/14/94

University of Washington

Abstract

An Adaptive Multilevel Method for Boundary Layer Meteorology

by David Eric Stevens

Chairperson of Supervisory Committee: *Professor Christopher Bretherton*
Department of Applied Mathematics

This thesis presents a new adaptive multilevel numerical model for incompressible flows and applies such a model for the first time to the simulation of atmospheric boundary layers capped by shallow cumulus and stratocumulus clouds. One needs to simulate a large domain, while locally having enough resolution to resolve the turbulent transports of mass, momentum, heat, and moisture in and around the clouds. Multilevel methods represent the flow on grids of varying resolution and are able to localize the high resolution and computational costs. In an adaptive multilevel method, the optimal regions of refinement are automatically determined and may evolve with time. The composite domain of an adaptive multilevel method is a dynamic balance between the need to resolve the global flow and regions of special interest.

In this work, a new forward-in-time multidimensional advection algorithm for incompressible flow with superior stability and accuracy is developed, analyzed and extended to a multilevel flow solver. By using object-oriented numerics and an efficient clustering algorithm, an adaptive numerical model was developed and used to analyze the transport properties of an isolated cumulus cloud. Examples of other forms of boundary layer convection that also benefit from adaptive multilevel modelling are presented for the first time.

TABLE OF CONTENTS

List of Figures

List of Tables

Acknowledgements

I wish to express my appreciation to Professors Chris Bretherton, Dale Durran, and Randy Leveque for their advice and guidance. I particularly thank Chris Bretherton for pushing me when I was right, advising me when I was wrong, and providing emotional support. I thank the University of Washington and the students, faculty and staff in the department of Applied Mathematics who have made this a very rich environment.

The research presented in this dissertation was supported in part by grants from the National Science Foundation (ATM-9216645, ATM-8858846).

To Leah and my mother.

1 Introduction

1.1 Boundary Layer Meteorology

Adaptive multilevel methods for incompressible flows have attracted interest as researchers seek to increase the resolution of their models. In many fluid flows, there are localized regions of fine-scale rapidly changing flow embedded in broad areas of more slowly varying motions. Multilevel methods represent the flow on grids of varying resolution and are able to localize the high resolution and computational costs. For many applications, it is not known a priori where high resolution will be required and fine scale features may move across the domain. An adaptive multilevel method is attractive for such applications. In an adaptive multilevel method, the optimal regions of refinement are automatically determined and may evolve with time. The composite domain of an adaptive multilevel method is a dynamic balance between the need to resolve the global flow and regions of special interest.

Numerical simulations of many atmospheric flows benefit from a multilevel approach. At the University of Washington, we have focused on cloud development in the atmospheric boundary layer (ABL). Two common cloud types in the ABL are isolated cumuli and stratocumulus cloud sheets. These clouds are being studied for their role in transport of heat and moisture and their effect on radiative transfer in the atmosphere. This transport is affected by the processes of latent heating, precipitation, condensation and evaporation. The cloud edge or interface between cloudy and unsaturated air is responsible for a large part of the turbulent mixing that occurs in a cloud and is crucial to the overall dynamics. It is desirable to resolve the cloud and its boundaries accurately. A cumulus capped ABL consists of turbulent clouds surrounded by expanses of stably stratified air that can be more coarsely resolved. One needs to simulate a large domain, while locally having enough resolution to resolve the turbulent transports of mass, momentum, heat, moisture and other substances in and around the clouds. For stratocumulus capped ABL's, it is important to resolve the finer scale eddies in an entrainment interface at the top of the

cloud layer, while convective turbulence within the rest of the layer is dominated by much larger eddy scales that don't require as much resolution.

1.2 New Numerical Methods

Recently developed "forward-in-time" advection schemes, which use information from only a single time level to advance, are becoming an increasingly popular tool for numerical solution of a variety of fluid flow problems, including atmospheric flows. These schemes allow simplifications in the timestepping of a multilevel method. They have improved stability, monotonicity, and conservation properties over traditional algorithms at no expense in accuracy. Meteorological models have traditionally relied on leapfrog schemes for their time differencing (Clark 1977; Sommeria 1975; Klemp and Wilhelmson 1978). Leapfrog schemes have the attractive feature of being non-dissipative, but are plagued with bad propagation characteristics and a computational mode. The temporal and spatial filters commonly used to control these defects adversely affect the accuracy of these schemes. Forward-in-time advection schemes have the equivalent accuracy of these schemes with none of their defects.

Forward-in-time schemes have been designed from several conceptual approaches to discretizing advection. Advection is the transport of fluid properties by fluid motions. Semi-Lagrangian schemes use this interpretation directly. In a semi-Lagrangian method, the value of an advected quantity at a gridpoint is equal to the value of the quantity at the previous time level at the departure point of a trajectory that ends at that gridpoint. A semi-Lagrangian method can be broken down into a trajectory calculation and an interpolation to determine the departure point. This breakdown simplifies analysis of the numerical properties of the method and has been used by Leonard (1993) and Smolarkiewicz and Pudykiewicz (1992) to help design and understand forward-in-time algorithms. Semi-Lagrangian methods are attractive because they are monotone, if a monotone interpolation is used, and because they are comparatively accurate and non-diffusive. If the trajectory can be computed accurately, they can be stable beyond Courant numbers of one. However, they are not conservative. One approach to designing conservative forward-in-time

schemes is to combine a subset of semi-Lagrangian ideas with Eulerian notions about fluxes through boundaries.

The usual conceptual approach to designing conservative advection schemes is the Eulerian finite volume interpretation that a grid point represents a cell whose tendency reflects the integral over the fluxes through the cell faces. An Eulerian method computes a flux as an integral over the characteristics that pass through a cell face during a timestep (Godunov 1959; LeVeque 1993; Bell 1989). It is desirable for the numerical method to enforce a discrete conservation law for the advected scalar on a cell-by-cell basis. This insures that for rapidly varying flows that the underlying conservation law is preserved, even though any numerical assumptions that are made about the continuity of the flow become invalid in the vicinity of sharp gradients.

The accuracy and monotonicity of an Eulerian method can be enforced by representing the characteristics to varying degrees of accuracy. LeVeque interprets this as the advection of different types of waves from one cell to the next. Over- and under-shoots can be thought of as over and under advection from one-cell to the next (Zalesak 1979).

1.3 Anelastic Approximation

Explicit numerical models are limited by the fastest propagating mode of the equations they model. In the unreduced equations, sound waves propagate much faster than the motions of interest. The actual Mach number of the flow is around 1/60. This requires a fully compressible explicitly time-differenced numerical model to limit its timestep by a factor of 60 compared to the Courant-Friedrich-Levy stability limit for advection alone. Time-split schemes that integrate the pressure-velocity coupling with a fast timestep limited by the sound speed and advection and forcing terms on a slow timestep limited by the advection are commonly used because they are fairly simple to implement, but are inefficient for boundary layer flows. Therefore, it is desirable to use an approximation to the governing equations in which the sound waves are filtered out. A system of equations commonly used in boundary layer meteorology are the original anelastic equations (Ogura

and Phillips 1962). These express the flow as a pseudo-incompressible fluid where the only effect of horizontal variations in density is to produce an apparent vertical force, the buoyancy. The scaling that leads to these equations is similar to another commonly used set of equations, the Boussinesq approximation. The primary assumptions required to derive the anelastic equations are that the deviation of the atmosphere from an isentropic base state is small, and that the Mach number of the flow is small. Others (Durran 1989; Lipps 1982) have relaxed the former assumption to improve the accuracy of their models for deep convection. However, the assumption of near isentropic flow is adequate for the shallow convection found in boundary layer clouds. We use the anelastic equations as the basis of our numerical model. The Boussinesq equations can be thought of as a limit of the anelastic equations, when the vertical extent of the fluid motions is much less than the density scale height of the fluid. Even for boundary layer flows of 1-2 km deep, this assumption is only marginally valid, and the anelastic equations more accurately represent the fluid flow.

Numerical methods for hyperbolic flows can be computed using a simple marching procedure. Methods that reintroduce compressibility, but with an artificial sound speed, are called artificial compressibility methods. This regains the hyperbolic character of the unreduced equations, while minimizing the loss of stability. These methods have been used in models for a number of years (Klemp and Wilhelmson 1978: Chorin 1967; Skamarock 1991; Durran 1983; Siems and Bretherton 1992). These methods use a time-splitting scheme similar to a fully compressible scheme, but require fewer fast timesteps due to the slower sound speed. Problems with separation of scales affect the accuracy of this treatment. To maximize efficiency, the artificial Mach number is typically 0.1-0.5, and the dispersion relations of gravity and sound waves changes significantly to the greatly reduced Mach number. This contrasts with the actual Mach number of around 1/60. The fast time scale can be thought of as an iterative elliptic solver for the pressure with each fast timestep corresponding to one iteration that advances the pressure toward its value at the next slow timestep. The artificial compressibility approach may not be as accurate and efficient as other elliptic solvers.

Another approach uses a diagnostic equation for the pressure. This approach dates back to the method of Harlow and Welch (1965). It splits the equations into a convective part and an elliptic pressure-velocity coupling. The convective part is handled by an explicit advection algorithm that computes the convective tendency. The pressure is computed as the quantity that maintains the continuity of the velocity field. This requires solving an elliptic Poisson equation every timestep. Fast direct and iterative elliptic solvers have been developed that are efficient enough to be used for a time-dependant model. This approach eliminates the need for short pressure-velocity timesteps and the associated accuracy problems, and fits nicely into a forward-in-time method.

1.4 Multilevel Methods

Multilevel methods are popular since they allow a model to resolve a wide range of scales. The computational expense is focussed at the localized fine scales, while the large scales have been retained. This thesis focuses on techniques that are characterized by the use of a block composite grid that is composed of a union of uniform fine grids. These grids are composed of regularly connected quadrilaterals that map very naturally on to the architecture of most computers. Each cell and its neighbors are manipulated by a very simple index space. These grids have the advantages of uniform grid discretizations: simple stencils, proven solvers, notions of accuracy, and computational efficiency. Non-block oriented multilevel schemes are more flexible to adapt to off-diagonal flows and arbitrary geometries, but suffer from a hard to manipulate index space and are hard to implement efficiently on most architectures.

In constructing a block composite method, one wants to create a comprehensive numerical treatment of the composite domain, while retaining the advantages and ease of use of uniform grids. McCormick (1989) formulated the finite volume element method (FVE) which is a combination of the finite volume method (FV) and the finite element method (FE). The finite volume method is used to create discretizations that are conservative and yield simple treatments of boundary conditions. The finite element method is used to determine the quadrature rules that are used in these discretizations and to develop

a theory of the accuracy of the resultant composite. It provides a methodology for discretizing at grid boundaries, but at the expense of creating irregular cells there. Most of the applications of this technique have been restricted to time-independent elliptic problems. This technique yields implicit solution techniques that become cumbersome especially when applied to time-dependent problems.

A multilevel method that retains to a larger degree the regular nature of a uniform grid is AMR (Adaptive Mesh Refinement). This was originally formulated for hyperbolic equations (Berger and Oliger 1984; Berger and Collela 1989) and has recently been extended to nearly incompressible flows (Almgren et al 1993). This technique is formulated about a cell-centered coordinate system where the discretization is based on a finite volume interpretation of the cells. There are no irregular cells on the borders between grids, so this method can be easily extended recursively to an arbitrary number of levels. This composite enables the use of standard uniform grid advection schemes. (Almgren et al implemented a Godunov-type scheme). The simple nature of this composite lends itself easily to complex systems of equations. The algorithm presented here is to combine the usefulness of AMR with ideas from FVE. This is done by using a refinement factor of three in the AMR composite. By using this refinement factor, the flexibility of the AMR composite for explicit forward-in-time algorithms and conservation between domains is retained, while taking advantage of the ideas of the FVE method.

The main difficulty in extending the AMR composite from hyperbolic to incompressible flows is the parabolic nature of these equations. In order to use this composite, a consistent formulation of conservation of mass must be constructed in a multilevel setting. One such formulation for a cell-centered discretization with staggered velocities was explored by Clark and Farley (1984). The internal boundary conditions for the normal velocity where found by a conservative interpolation technique. This thesis uses a formulation of this technique. It has the advantage of being able to use a fast direct FFT based solver for computing the pressure. Another technique for formulating mass conservation is

described by Almgren et. al. (1993). This technique uses a multilevel solver on the full composite grid to insure mass conservation.

Several compressible nonhydrostatic meteorological models (Klemp and Wilhelmson 1978; Skamarock 1991) are routinely used in a multilevel setting. Skamarock has demonstrated an adaptive multilevel compressible code which was used to simulate density currents and severe convective storms. Because the equations are purely hyperbolic, treatment of grid interfaces is simpler. However, the time-split algorithm is not conservative and requires smoothing to make stable in addition to being inefficient. The anelastic AMR scheme presented here overcomes these difficulties.

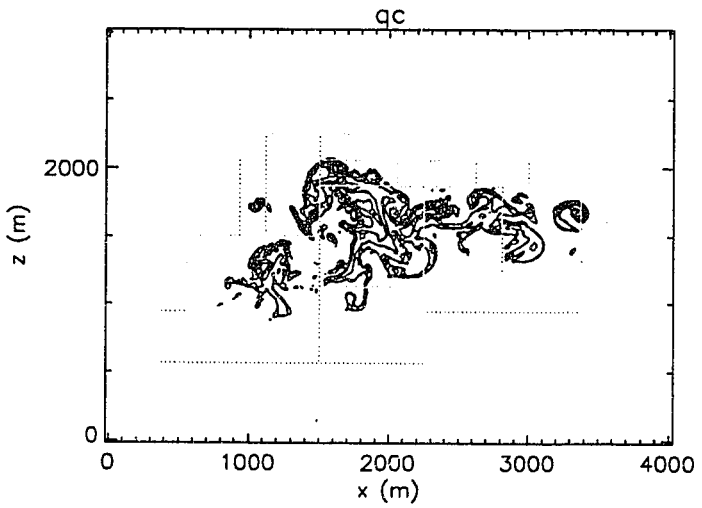

Figure 1.1: Illustration of a Cumulus cloud simulated with self-adaptive refinement, with currently refined domains dashed. This illustrates the ability of an adaptive multilevel method to conserve both computational and memory expenses.

An adaptive algorithm that adjusts the composite domain as the solutions evolves is important to providing the resolution required by an evolving flow. There are several techniques for determining the location of refinement (Berger and Collela 1984; Berger and Rigoutsos 1989). A common approach (Clark and Farley 1984) is to use information from

previous simulations and recompute the solutions using fixed domains. This is a cumbersome and inefficient manual approach that applies best to flows forced in a fixed spatial location. Another approach is to make some estimate of the local error and refine in such a way as to keep this estimate below a certain tolerance. This estimate in an AMR algorithm is commonly found using Richardson extrapolation. This requires a knowledge of the overall accuracy of the algorithm to compare a coarse and fine solution. If an algorithm is pth order accurate and the fine solutions is twice as refined as the coarse, an estimate of the fine grid error is given simply by

$$E = \frac{U_{h/2}\left(x, t^n\right) - U_h\left(x, t^n\right)}{2^p - 1} \tag{1.1}$$

This is a natural indicator for a multilevel method, since the solution is often provided on both coarse and fine levels. This provides no initial information on refining the coarsest level, but can be patched by using an even coarser grid or by using another estimate to start the refinement process. This error estimation may break down in regions around sharp gradients, where any error estimates that rely on continuity of the solution are invalid. Other problem-specific quantities may be used as error indicators. The deformation of the flow is an indicator of unresolved turbulence and mixing. The presence of liquid water is a good choice of an error indicator for simulations of moist convection, where turbulence is localized in clouds where there is a large release of latent heating.

Once the regions where refinement are needed are defined, they need to be clustered into computationally efficient domains. Block refinement can be made using grids aligned with the coarse domain or rotated. A non-rotated algorithm was developed by Berger and Rigoutsos (1989). This algorithm creates refinement by analyzing histograms of cells that have been marked for refinement. Natural breaks or gradients in these histograms are designated as natural places to subdivide the domain. This technique has the advantage of creating internal domains that are non-overlapping. We will explore a variation of this technique. Nonrotated algorithms have the benefit of creating composites that can use discretizations of the FVE and AMR type to insure accuracy and conservation. These algo-

rithms suffer from being cumbersome in refining features that are aligned oblique to the coarse grid. Examples of algorithms with rotated grids have been presented by Skamarock (1993) and Berger and Oliger (1984). Rotated grids are efficient in refining oblique features with a minimal number of refined domains. Rotated grids suffer from being hard to analyze, nonconservative, and often produce regions of overlapping refined domains where work is duplicated.

1.5 Object-Oriented Numerics

Algorithmic advances have been helped by advances in computer science for managing the complexity that naturally arises in a multilevel scheme. The multilevel method described in this thesis was written in an object-oriented fashion using C++. This type of programming allows the user to construct classes which represent abstractions like advection schemes and array operations. This provides the programmer effective complexity control not present in a procedural language like FORTRAN or C. By linking functions with the data that are processed by them, one can eliminate the tedium of always working at the lowest level. The lack of these tools in a procedural language make many tasks difficult to tackle. This multilevel method would not be a feasible project for a student without these tools.

An important part of these abstractions is that their state is controlled by a well-defined interface. By being able to control the creation, destruction, and manipulation of various states in the code, one gains the ability to decompose a complex code into independent blocks each with its own consistent, robust logic. This creates the redundancy needed to create and maintain a complex code. An object-oriented language is a natural choice for implementing an adaptive multilevel method. If one can create a composite domain initially, adaptively modifying it becomes a well-defined process of adjustment and reinitialization with little more complexity.

1.6 Boundary-Layer Cumulus Convection

Shallow cumulus convection plays a key role in the transport of heat and moisture in the atmosphere. Boundary layer clouds and shallow cumulus clouds can be instrumental in transporting moisture to a level at which it is detrained into broad cloud sheets. This has an important and poorly understood effect on the earth's radiation budget. To examine these global impacts of shallow cumuli, we must incorporate their effects into large-scale numerical models in which none of this convection is resolved.

Cumulus parameterization is the art of creating simple models of cumulus convection for use in these large-scale models. Arakawa and Schubert (1975) modelled cumulus convection as an ensemble of cumulus clouds each with their own constant fractional rate of lateral entrainment. A 'buoyancy-sorting' model of a cumulus cloud by Raymond and Blyth (1986) has attracted attention recently, and inspired the Emanuel (1991) cumulus parameterization. This model treats cumulus convection as an ensemble of mixtures of cloud base and environmental air that rise or fall to their level of neutral buoyancy. Each of these models involves different assumptions about the relevant behavior of cumulus clouds. There is still much controversy about the best conceptual model of the mass, momentum, heat and moisture transports in a single cumulus cloud or cumulus ensemble. Numerical simulations and observations of shallow cumuli can be coupled to provide a powerful tool for determining how these transports are determined by large scale conditions. Much of the observational and modelling effort in studying the dynamics of these clouds has as its goal improving these simple models.

A goal of cumulus convection studies is the determination of the mechanisms by which entrainment of surrounding air into a cumulus cloud and mixing of air within the cloud occur and how they affect the evolution of the cloud. One long-standing question is whether entrainment is primarily driven by penetrative downdrafts caused by buoyancy reversal, by a baroclinic instability of the cloud-environmental air interface, or by lateral entrainment. Buoyancy reversal is caused by the evaporative cooling that occurs when buoyant moist cloudy air mixes with dry environmental air and produces a mixture that is

more negatively buoyant than any of its initial components. The Raymond and Blyth model uses this mechanism to predict the evolution of a cloud given a fixed entrainment profile. Baroclinic instabilities are caused by denser fluid overlying buoyant fluid and the fluid trying to overturn itself.

The hypothesis that penetrative downdrafts caused by buoyancy reversal are responsible for cloud top entrainment was proposed as far back as Telford (1975). This hypothesis was popularized by Paluch (1979), who observed for a continental cumulus cloud that a large fraction of the observed air appeared to be composed of mixtures of cloud base and cloud top air. Her study introduced a modified mixing diagram analysis that is widely used to analyze the origins of cloudy air. This hypothesis is supported by the fact that the maximum amount of buoyancy reversal that can occur is from between these two levels. Siems and Bretherton (1992) further explored the effect of buoyancy reversal by large eddy simulation.

A baroclinic mechanism of entrainment was investigated by the modelling study of Klassen and Clark (1985) and Grabowski and Clark (1991). They hypothesize that the shear flow of dry environmental air being forced over moist cloudy air was responsible for the formation of downdrafts around the edges of the cloud or through them. Small perturbations were found to change the nature of the cloud top from forced descent around the edges to penetrative downdrafts in the center. These modelling studies used the multilevel method of Clark and Farley (1984). This method enabled them to adequately resolve the cloud and the small scales that are important to the entrainment process. However, the studies did not quantitatively consider how much air from different levels was entrained into the cloud or what the "bulk" effect of entrainment was.

Another focus of cumulus convection studies is to determine what the bulk effects of the entrainment processes are in the presence of an environmental forcing. Observational studies of the HARP project were made to compute estimates of the bulk properties. Grinnell et al (1995) used dual Doppler measurements to construct composite mass flux profiles of various combinations of Hawaiian cumuli. Raga et al (1990) constructed mass and

moisture budgets for a composite Hawaiian cloud. They used Paluch analysis to analyze the origin of air in these clouds. The numerical study by Bretherton and Smolarkiewicz (1989) used a tracer technique to diagnose the mass fluxes inside of a continental cumulus cloud. This technique uses a tracer of horizontal position to diagnose the amount of mass that was entrained at this level. The results qualitatively supported the model of Raymond and Blyth (1986); entrainment occurred fairly uniform from all heights while detrainment occurred preferentially in stable layers of the environment.

These bulk effects of a field of shallow cumuli were also analyzed for an ensemble of cumuli in a trade wind boundary layer in a large-eddy simulation study by Sommeria (1975). He observed the sensitivity of the cloud base mass flux to the thermodynamic conditions of the cloud layer and evaluated the thermodynamic deviations of the clouds from the environment. He was able to construct heat and moisture budgets for the trade wind boundary layer. A limitation of this study was that it offered little insight into how the structure of individual clouds was related to the overall fluxes, a key issue in cumulus parameterization.

This thesis uses a multilevel method to simulate cumulus convection in a trade wind boundary layer and to create vertical entrainment and mixing profiles. The multilevel method enables the simulation to better resolve the cloud scale motions that govern the entrainment. We seek to analyze the effect of a single cumulus cloud on the environment and from the bulk effects to shed light on the entrainment mechanisms that govern this effect. The response of the cloud to different environmental conditions is examined. Varying relative humidity in the cloudy layer affects the amount of evaporative cooling that can occur. This should affect dynamics caused by buoyancy reversal. Varying shear in the cloudy layer affects the dynamics around the cloudy-environmental air interface. This should affect dynamics governed by shear and baroclinic processes.

1.7 Thesis Goals and Organization

In this thesis, we develop a new forward-in-time numerical method for adaptive multilevel modelling of incompressible flow for boundary layer meteorology. The numerical method is based on a new forward-in-time advection algorithm that is second order accurate for variable source terms and solenoidal velocity fields. This algorithm combines and improves upon many features of recent work in the numerical analysis of incompressible flow to create an efficient, monotone and stable algorithm. The application of this algorithm to the anelastic equations is explored and the numerical scheme is verified using advection tests, conservation of energy and comparison with a linear gravity wave solution. The full model is extended to a multilevel framework with a complex composite domain consisting of a block composite grid with nested grids aligned along the coordinate directions. This code is implemented in C++ using object-oriented numerics. This allows us to create a powerful and flexible numerical model. Special care is taken to both minimize computational and memory requirements. We then use this model to analyze the entrainment, detrainment, and mixing that occurs in an isolated cumulus cloud, and to compare how they depend on the environmental sounding with simple conceptual models of cumuli. Lastly, we show preliminary examples of other applications of this model to stratocumulus-capped boundary layers and cumulus cloud fields.

The physical problem is presented in the first chapters. Chapter 2 introduces the basic equations of motion used, including their extensions to atmospheric boundary layers and trade cumuli. Chapter 3 derives a new forward-in-time algorithm from both a semi-Lagrangian and Eulerian perspective. This algorithm is extended in Chapter 4 to the full anelastic system of equations. We use the response of a heat source in stable stratified flow as a nonlinear test of the model and its conservation properties.

The earlier chapters are extended in chapters 5 and 6 to a multilevel setting. Various methods of enforcing conservation across grid interfaces are examined spatially and temporally in Chapter 5 and our multilevel algorithm is described. Chapter 6 introduces

object-oriented numerics and illustrates its usefulness in creating a multilevel composite domain.

Trade cumulus convection as an application of the model is analyzed in Chapter 7. We focus on the properties of trade cumulus convection as derived from analysis of derived entrainment, detrainment and mixing. The effect of environmental shear and varying relative humidity on the cumulus convection is examined as well as the effect of the cumulus convection on the thermodynamic structure of the atmospherics sounding. A comparison with radar observations of trade cumuli near Hawaii is also discussed.

In Chapter 8, we summarize the results of the previous chapters. The adaptive multilevel method is able to increase the effectiveness of a numerical model and its application to cumulus convection. We seek to connect our research with other current work and provide future directions of research. A growing body of object-oriented software and results from parallel computing and domain decomposition are rapidly developing multilevel numerical algorithms. Many meteorological problems involve flow over terrain and nonlocal physics such as radiation that present challenges for multilevel numerics. Multilevel methods have further promise for improving our understanding of atmospheric convection. We provide an example from stratocumulus convection of the entrainment of environmental air above the top of a stratus deck. Multilevel methods have promise for simulating an actively convecting trade wind boundary layer and regions of mesoscale convection in the tropics.

2 Modelled Equations

2.1 Anelastic Equations

The "anelastic approximation" to the equations of motion is commonly used in nonhydrostatic modelling of atmospheric convection. In this study, we use the original anelastic approximation of Ogura and Phillips (1962). Other approximations (Lipps and Hemler 1982; Durran 1989) for low Mach number flows are also commonly used. They have similar accuracy when applied to flows in which all fluid parcels have comparable densities when moved adiabatically to a reference pressure, but may be more accurate for highly stratified flow (e.g. Durran 1989). The advantage of the original anelastic equations is their simplicity as well as the fact that they preserve analogies to all the conservation laws of the full compressible equations. These equations can be derived from the Euler equations

$$\begin{aligned} \frac{1}{\rho}\frac{d\rho}{dt} + \nabla\bullet\vec{U} &= 0, \\ \frac{d}{dt}\vec{U} &= -\frac{\nabla P}{\rho} - g\hat{k}, \\ \frac{d\theta}{dt} &= 0, \end{aligned} \tag{2.1}$$

where conservation of internal energy has been expressed using potential temperature

$$\theta = T/\pi = T\left(\frac{P}{P_o}\right)^{-R_d/C_{pd}}, \qquad (P_o = 1000\text{mb}).. \tag{2.2}$$

Here R_d and C_{pd} are the gas constant and specific heat of dry air, respectively. The potential temperature is the temperature obtained by adiabatically moving a parcel of dry air to a reference pressure. This temperature is related to the entropy of a parcel of dry air and is conserved with adiabatic motion.

We will follow the anelastic assumption that the relative deviation of θ from an isentropic base state, θ_o, is small everywhere in the layer. If a typical deviation of θ is $\Delta\theta$, the small parameter

$$\varepsilon = \frac{\Delta\theta}{\theta_o} \tag{2.3}$$

can be used to scale the equations. This assumption implies that buoyant accelerations in a gravitational field with downward gravitational acceleration g are all $O(g\varepsilon)$ or smaller and sets a minimum timescale for such accelerations to create displacements of $O(H)$:

$$\tau = N^{-1} = \left(\frac{H}{g\varepsilon}\right)^{1/2}. \tag{2.4}$$

where H is the scale height. If the variations in θ are mainly due to the mean stratification, N is the mean buoyancy frequency within the layer. Note that the advective velocity scale

$$U_o = H/\tau = (gH\varepsilon)^{1/2}, \tag{2.5}$$

is much less than the sound speed $c_s = (gH_s)^{1/2}$, if ε is small and the layer depth H is comparable to or less than the pressure scale height H_s which is $O(8\text{km})$ for the atmosphere. For boundary layer flows with $\Delta\theta \leq 10\text{K}$ and $H \leq 2\text{km}$, the characteristic Mach number $M = U_o/c_s$ is given by

$$M = H/(\tau c_s) \leq 0.07 \ . \tag{2.6}$$

The anelastic equations are analogous to the even simpler Boussinesq equations, in that the effect of buoyancy is retained, but sound waves are filtered out. The Boussinesq equations apply to low Mach number flow in a domain with $H \ll H_s$ and $\varepsilon \ll 1$. The anelastic equations apply to the same type of flow, but can be used for a deep domain with larger vertical variation in the mean density, so long as relative potential temperature variations are small over the entire domain depth.

Following Ogura and Phillips (1962) one can transform the momentum equation, by non-dimensionalizing the variables and replacing P and ρ by θ and π.

$$\varepsilon\beta\frac{d}{dt}\vec{U} = -\theta\nabla\pi - \beta\hat{k}. \tag{2.7}$$

The dimensionless number that arises,

$$\beta = \frac{gH}{C_{pd}\theta_o}. \tag{2.8}$$

is the ratio of the domain depth to the depth of an isentropic atmosphere and is a measure of the depth of the convection. The scaling assumption (2.3) is applied by expanding all of the dependant variables in the momentum equation as power series in ε:

$$\begin{aligned} \theta(\hat{x}, t) &= 1 + \varepsilon\theta_1(\hat{x}, t) + \ldots, \\ \pi(\hat{x}, t) &= \pi_o(\hat{x}, t) + \varepsilon\pi_1(\hat{x}, t) + \ldots, \\ \vec{U}(\hat{x}, t) &= \vec{U}_o(\hat{x}, t) + \varepsilon\vec{U}_1(\hat{x}, t) + \ldots, \end{aligned} \tag{2.9}$$

The zeroth order momentum equation reveals that the base state pressure is hydrostatically balanced with the vertical structure of the isentropic base state, $\pi_o = 1 - \beta z$. The first order momentum equation yields the dynamical balance,

$$\beta\frac{d\vec{U}_o}{dt} = -\nabla\pi_1 + \beta\theta_1. \tag{2.10}$$

A feature of the anelastic momentum equation is that the buoyancy term $\beta\theta_1$ does not include the effect of the buoyant acceleration caused by the pressure perturbation. This effect has been incorporated into the modified pressure gradient term $-\nabla\pi_1$.

The equations that are integrated in the model are formulated in terms of this expansion, where for an arbitrary dimensional variable ψ, the decomposition,

$$\psi = \langle\psi\rangle\psi_o + \varepsilon\langle\psi\rangle\psi_1 = \Psi + \psi' \tag{2.11}$$

is used where $\langle\psi\rangle$ is the scaling for ψ. The momentum equation in its dimensional form becomes

$$\begin{aligned}\frac{d}{dt}\vec{U} &= -C_{pd}\theta_o\nabla\pi' + g\frac{\theta'}{\bar{\theta}}\hat{k}, \\ &= -\nabla\phi + g\theta^*\hat{k}.\end{aligned} \tag{2.12}$$

where $\phi = P'/\bar{\rho} = -C_{pd}\theta_o\pi'$ is the pressure perturbation scaled by the mean density and $\theta^* = \theta'/\bar{\theta}$ is the relative potential density perturbation. The buoyancy with respect to the reference is just $g\theta^*$. It satisfies the equation

$$\frac{d}{dt}\theta^* = 0. \tag{2.13}$$

In the unreduced system of equations, sound waves propagate much faster than the fastest motions of interest. Even though most of the energy is present in the gravity wave modes, the timestep is reduced greatly by the need to propagate the acoustic modes. Using the compressible system of equations (2.1) is therefore inefficient. The anelastic scaling assumptions lead to a mass continuity equation that filter out sound waves. The first term in the unreduced continuity equation can be written as:

$$\frac{1}{\rho}\frac{d\rho}{dt} = -\frac{wg}{c_s^2} + \frac{1}{\rho c_s^2}\frac{d}{dt}P', \tag{2.14}$$

where C is the sound speed and $\rho = \bar{\rho} + \rho'$. By using the scaling relationship

$$\frac{U^2}{c_s^2} = \varepsilon\frac{H}{H_s} \ll 1, \tag{2.15}$$

we can justify neglecting the P' term. The zeroth order anelastic continuity equation that results from this assumption is

$$\nabla\bullet\bar{\rho}\vec{U} = 0. \tag{2.16}$$

To close the system of equations (2.12), (2.13) and (2.16), a diagnostic relationship is needed for the scaled pressure perturbation, ϕ. This relationship is found by computing the divergence of the momentum equation and noting that

$$\nabla \bullet \bar{\rho} \frac{\partial \vec{U}}{\partial t} = 0 \,. \tag{2.17}$$

The diagnostic relationship is an elliptic equation

$$\nabla \bullet \rho \nabla \phi = -\nabla \bullet (\rho (\vec{U} \bullet \nabla) \vec{U} + \rho g \theta^* \hat{k}) \,, \tag{2.18}$$

that must be solved every timestep. The dynamic pressure adjusts instantaneously so as to enforce continuity. The elliptic equation was simplified into a form of Poisson equation by our manipulation of the momentum equation into a form where the buoyancy depends only on the potential temperature deviation $g\theta^*$.

2.2 Atmospheric Convection

The discussion so far ignores moisture and sources of heat and momentum. Moisture affects the environment due to the latent heat release of condensation and the effect of moisture on air density, both of which are important in the boundary layer. The effect of moisture on air density is due both to the different densities of water vapor and dry air, and due to the gravitational loading of suspended liquid water droplets. This requires the buoyancy in the anelastic equations to be redefined in terms of virtual potential temperature,

$$\theta_v = \theta (1 + 0.61 q_v - q_c) \tag{2.19}$$

rather than θ, where q_v and q_c are the mixing ratios of water vapor and liquid water to dry air.

To account for the vertical structure of the atmosphere superimposed on the isentropic base state, we expand the variables further

$$\begin{aligned}
\pi(\vec{x},t) &= \bar{\pi}(z)+\pi'(z)+\pi''(\vec{x},t) = \pi_{env}(z)+\pi''(\vec{x},t),\\
\theta_v(\vec{x},t) &= \bar{\theta}+\theta'_v(z)+\theta''_v(\vec{x},t) = \theta_{venv}(z)+\theta''_v(\vec{x},t),\\
q_v(\vec{x},t) &= q'_v(z)+q''_v(\vec{x},t) = q_{ve}(z)+q''_{venv}(\vec{x},t),\\
q_c(\vec{x},t) &= q'_c(z)+q''_c(\vec{x},t) = q_{ce}(z)+q''_{cenv}(\vec{x},t),
\end{aligned} \tag{2.20}$$

as in Klassen and Clark (1985). The barred terms are those of the isentropic base state, the primed terms are the deviation of the static environment from the base state, and the double prime terms are the dynamic perturbations of the environment. By scaling the momentum equation as in the derivation of the anelastic equations and assuming that the dynamic perturbations are small compared to the environmental deviation, the environmental deviation is found to be in hydrostatic balance.

$$C_{pd}\bar{\theta}\frac{\partial\pi'}{\partial z} = \frac{g\theta_v'}{\bar{\theta}} \tag{2.21}$$

The momentum equation (2.12) can be written

$$\bar{\rho}\frac{d}{dt}\vec{U} = \bar{\rho}C_{pd}\bar{\theta}\nabla\pi'' + \bar{\rho}\frac{\theta_v''}{\bar{\theta}}\hat{k} \tag{2.22}$$

Note that buoyancy $B = g\theta_v''/\bar{\theta}$ and the dynamic pressure perturbation π'' involves only perturbations from the mean environmental sounding.

Condensation has an important effect on the dynamics of boundary layer convection, due to latent heating. It is common to partition liquid water into 'cloud' and 'rain' droplets. The cloud droplets are a few microns in size and are assumed to have a negligible fall speed and evaporate instantaneously where the relative humidity is less than unity. Raindrops fall and evaporate at finite rates. In this thesis, we will examine only shallow convection in which precipitation processes are usually negligible. Thus we will assume all liquid water is in the form of cloud droplets. In this case, the thermodynamic state can be expressed with two variables that are conserved during adiabatic motion: the liquid water potential temperature θ_l and total water mixing ratio q_t

$$\theta_l = \theta - \frac{Lq_c}{C_{pd}\pi_{env}}, \qquad q_t = q_v + q_c, \tag{2.23}$$

where T_{env} is the environmental temperature. θ_l has the advantage of reducing to potential temperature θ in the absence of liquid water. The motion of these two variables is described by a tracer equation in the form of

$$\frac{d\psi}{dt} = S_\psi \tag{2.24}$$

In practice, it is advantageous to advect the scaled deviation of θ_l from the isentropic base state

$$\theta_l^* = (\theta_l - \bar{\theta})/\bar{\theta}. \tag{2.25}$$

Condensation is assumed to occur instantaneously when the total mixing ratio exceeds the saturation mixing ratio. This approach eliminates the need to advect both q_v and q_c simultaneously. An estimate of the temperature perturbation is required for determining the saturation mixing ratio of a volume. This temperature perturbation is related to the potential temperature perturbation with respect to the environment by,

$$T_{env}^* = \frac{T''}{T_{env}} \approx \frac{\theta''}{\theta_{env}} \tag{2.26}$$

Computing T_{env}^* from environmental values and neglecting the dynamic pressure perturbation results in a small error, whereas in the notation of Ogura and Phillips neglecting the π' term in

$$\frac{T'}{T} = \frac{\theta'}{\theta} + \frac{\pi'}{\pi} \tag{2.27}$$

results in significant error due to the deviation of the environment from an isentrope (Klassen and Clark 1985).

Since $\theta_l = \theta$ for dry air, computing condensation is simple. We set $q_c = 0$ and $\theta = \theta_l$ and test for saturation. If the air is unsaturated, our assumption is consistent.

$$q_{vs}(T^*_{env}) \le q_t \Rightarrow \begin{cases} q_v = q_t \\ \theta = \theta_l \\ q_c = 0 \end{cases} . \tag{2.28}$$

Here $q_{vs}(T^*_{env})$ is the saturation mixing ratio of air relative to the saturation mixing ratio of the environment q_{vsenv}. The Clausius-Clapeyron equation is used to determine q_{vsenv} via

$$q_{vsenv} = \varepsilon \frac{e_s(T_o)}{P_{env}} \exp\left(\frac{L}{R_v T_o} \left(\frac{T_{env} - T_o}{T_{env}} \right) \right) \tag{2.29}$$

If $q_{vs}(T^*_{env}) > q_t$, a volume is saturated. In this case, an estimate of T^*_{env} is required for determining the saturation mixing ratio of a volume. T^*_{env} is related to θ_l and q_c, by the definition of θ_l

$$q_c = \frac{C_{pd} \pi_{env} T_{env}}{L} (\theta_l - \theta_{env}(1 + T^*_{env})) \tag{2.30}$$

and to q_t by the relationship

$$\begin{aligned} q_c &= q_t - q_{vs}(T^*_{env}) \\ &= q_t - \left(\frac{q_{vsenv}}{1 + T^*_{env}} \right) \exp\left(\frac{L}{R_v T_{env}} \left(\frac{T^*_{env}}{1 + T^*_{env}} \right) \right) \end{aligned} \tag{2.31}$$

An implicit relationship for T^*_{env} in terms of the model variables θ_l and q_t is found by equating (2.30) and (2.31). This relationship can be solved iteratively by Newton's method. Usually, three iterations are enough for convergence.

2.3 Trade Cumuli

The vertical thermodynamic structure of the atmosphere determines whether the release of latent heat in rising cloudy air overcomes the ambient stable stratification of θ in the surrounding unsaturated air, rendering the atmosphere conditionally unstable to moist convection. While the most spectacular manifestation of moist convection is perhaps a severe thunderstorm extending 10-20 km above the ground, we will consider here another form of convection which is important, because it is ubiquitous over vast areas of the subtropical and tropical oceans, the trade cumulus boundary layer. In the following example, we consider a slightly buoyant bubble in a vertical "sounding" (θ and q_t profile) typical of a trade cumulus boundary layer over the tropical oceans. This sounding (Figure 2.1:) is an idealization of soundings taken near Hawaii. The example illustrates the nonlinear effects on buoyancy that our treatment of condensation in Section 2.2 introduces.

A typical trade cumulus boundary layer can be thought of as being composed of a sub-cloud mixed layer, a cloudy layer that is conditionally unstable and a trade inversion. The trade inversion is strongly stably stratified and prevents convection from piercing it. These features are dynamically maintained by the interaction of the cumulus convection with the dry, warm subsiding airflow above the boundary layer. A conceptual model of the typical lifecycle of a trade cumulus cloud starts with a "bubble" of air in the sub-cloud layer which is slightly buoyant, it rises to the top of this layer, where the water vapor starts to condense into cloud. The bubble, now a cumulus cloud, continues to rise due to the generation of latent heat by condensation; and is typically quite turbulent. The cloud is stopped at the trade inversion and evaporates by mixing with the surrounding air.

Figure 2.2 illustrates the effect of condensation on an air parcel being adiabatically lifted in the sounding of Figure 2.1. The parcel originates at an altitude of 300 meters in the sounding with a θ perturbation of 0.5 K and a q_t perturbation of 0.3 g/kg. These perturbations are representative of perturbations below trade wind cumuli. The parcel

becomes strongly buoyant as it enters the cloudy region at 600 meters and will rise to a height of 2100 meters if no mixing occurs.

A small amount of entrainment (turbulent mixing incorporating environmental air) can have a large effect on the trajectory of an air parcel. Notice that the buoyancy B is a non-linear function in terms of the conserved variables θ_l^* and q_t.

$$B = g\left(\theta_l^* - \theta_{lenv}^* + \frac{Lq''_c}{C_{pd}T_{env}\theta_o} + \varepsilon_o q''_v - q''_c \right) \tag{2.32}$$

because it depends on both q_c and q_v. When air parcels turbulently mix, θ_l^* and q_t vary linearly with the mixing fraction, but B does not. In particular, if a clear and cloudy air parcel mix, some of the liquid water in the cloudy parcel evaporates, reducing B.

Evaporation due to mixing of environmental air has an important effect on the final destination height of a mixed parcel. Figure 2.3 and Figure 2.4 show this effect by considering mixing of the parcel in Figure 2.2 with various proportions of ambient air at different heights, a process that naturally occurs at the turbulent cloud edges. Figure 2.3 shows the initial θ_v'' perturbation of the mixed parcels. At the inversion, the mixing of warm dry environmental air causes most mixtures to be negatively buoyant. The most negatively buoyant mixtures are those which entrain just enough environmental air to evaporate all liquid water. The destination height z_{dest} is the level to which a mixture rises or sinks before it becomes neutrally buoyant with respect to the ambient air at that level. Figure 2.4 shows that parcels can often travel far from their level of mixing. Most mixtures at the inversion drift downwards. Most mixtures at cloud base rise to near the inversion. At middle levels, a small shift in the environmental air can determine whether a mixture rises to near the inversion or only a few hundred meters.

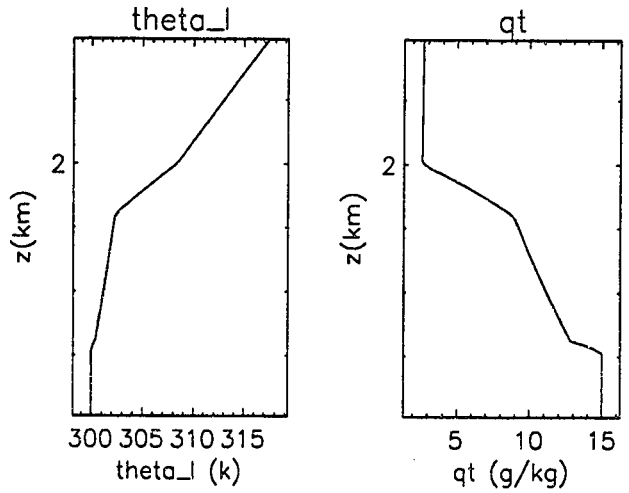

Figure 2.1: Soundings of potential temperature and moisture, typical of the trade cumulus boundary layer on the windward coast of Hawaii. The sounding has a cloudy layer that starts around 700 meters and an inversion around 1800 meters.

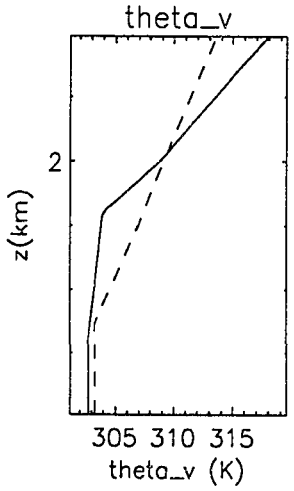

Figure 2.2: Environmental sounding of virtual potential temperature and the virtual potential temperature of a parcel on an adiabatic trajectory for the sounding of Figure 2.1. The solid line is the environment and the dashed is the parcel.

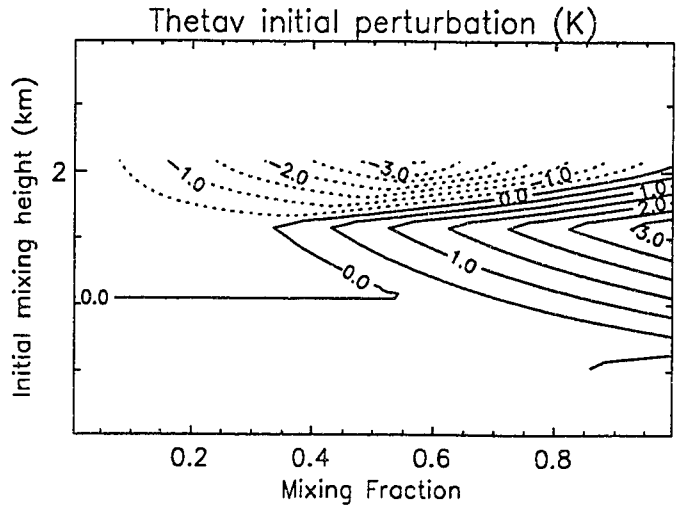

Figure 2.3: Initial θ_v perturbation of mixed parcels. The ordinate is the height at which the parcel is mixed and the abscissa is the fraction of undilute cloud base air. The perturbation has units of K and the contour interval is 0.5 K.

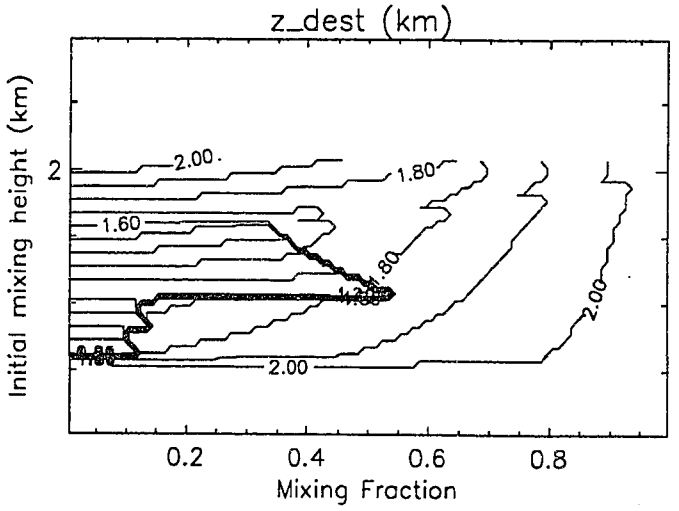

Figure 2.4: Final destination of mixed parcels. The final destination is shown with units of km and a contour interval of 0.2 km.

3 Advection

3.1 Introduction

We are interested in modelling the transport properties of cumulus convection in the trade wind boundary layer as described in Chapter 2. This requires an accurate numerical algorithm for advection as described by the transport equation of a scalar ψ,

$$\frac{\partial}{\partial t}\rho\psi + \nabla\bullet\rho\psi\vec{U} = \rho R, \tag{3.1}$$

with source term R and velocity field $\vec{U}$ in a quasi-incompressible fluid with the continuity equation,

$$\nabla\bullet\rho\vec{U} = 0 \quad . \tag{3.2}$$

This chapter derives and tests an Eulerian forward-in-time advection algorithm that models (3.1) and is used in all the transport processes of the full numerical model. In Chapter 5, this algorithm will be generalized to a multilevel framework. A multilevel method is defined as a domain decomposition method where the solution is represented on a composite domain of refined meshes. A forward-in-time advection method is a method that uses only the most recent time level to advance the solution in time. This is in contrast to other advection methods that are commonly called multilevel methods, such as leapfrog. In that context, multilevel refers to the use of two or more time levels to advance the solution in time.

An attractive feature of forward-in-time methods is their ease of use. Because only one time level is used, it is not necessary to keep previous solutions in work arrays. This decreases the storage requirements considerably and eliminates the need of special start-up procedures. These methods are also simply generalizable to an adaptive multilevel framework, because the communication between grids is kept to a single time level. This will be seen in chapter 4 to facilitate the conservation of scalars across grid interfaces.

Our forward-in-time algorithm uses ideas from both semi-Lagrangian and Eulerian interpretations of what a gridpoint represents. The Eulerian perspective is that a gridpoint represents a cell average centered about the gridpoint's physical location. Its time tendency is determined by the fluxes that cross the cell boundary. A gridpoint in a semi-Lagrangian method represents the endpoint of a trajectory and advection is computed by analyzing this trajectory. Eulerian forward-in-time algorithms have discrete conservation properties. This ensures that numerical solutions stay faithful to the original conservation laws that they approximate.

Ideally, an advection algorithm should be both monotonic and not introduce numerical dispersion or dissipation into the flow. Monotonicity is necessary to prevent numerical overshoots that may affect the accuracy of source terms that depend on advected quantities. A dispersive algorithm changes the dispersion relationship of the flow, whereas a diffusive algorithm excessively blurs the flow. We use an Eulerian flux correction technique to make this algorithm monotonic with minimal loss of accuracy. Flux correction helps minimize the effects of a dispersive method by damping out spurious oscillations caused by sharp gradients. The flux correction is turned off in smooth regions of the flow to avoid adding excessive diffusion.

It is important that the flow preserves the symmetries that are present in the flow. A nonlinear advection algorithm (Smolarkiewicz 1986) can introduce asymmetries into the advection of velocities which are symmetric about zero. An advection algorithm should not introduce a directional bias for flows which are oblique to the grid. The numerical simulation of the turbulence that occurs in a cumulus cloud should not be given an anisotropic bias by the alignment of the grid. This chapter uses a semi-Lagrangian interpretation to provide a natural way of creating discretizations that accurately represent these oblique flows.

3.2 Derivation

Semi-Lagrangian algorithms treat the transport equation in its convective form,

$$\frac{D\psi}{Dt} = R. \tag{3.3}$$

Rather than looking at the fluxes through a grid cell to determine the tendency of a volume, the semi-Lagrangian approach of advancing forward in time is to determine the trajectory that intersects a gridpoint at time n+1 and determine the value of the solution at its departure point at time level n. In the absence of sources, this is expressed as

$$\psi\left(\hat{x}, t^{n+1}\right) = \psi\left(\hat{x}_o, t^n\right), \tag{3.4}$$

where $\left(\hat{x}, t^{n+1}\right)$ is the gridpoint at n+1 and $\left(\hat{x}_o, t^n\right)$ is the departure point at the start of the trajectory. The accuracy of a semi-Lagrangian method depends on the accuracy of the trajectories and the accuracy of the interpolation procedure required to get $\psi\left(\hat{x}_o, t^n\right)$ from gridpoint values at time t^n. If the velocity is constant, the trajectories are straight lines and the monotonicity and accuracy properties depend on the interpolation algorithm, as indicated by Figure 3.1. The trajectory T connects levels n and n+1 and the interpolation algorithm determines the value of the solution between i and i-1.

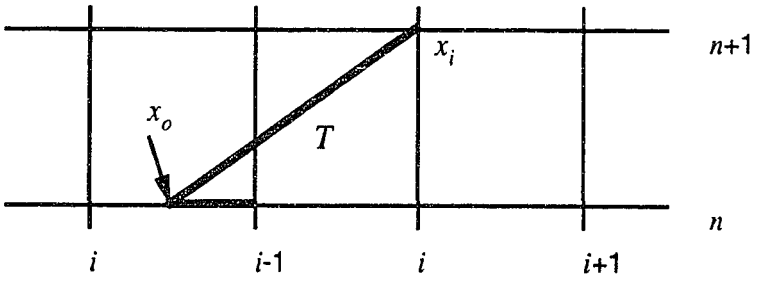

Figure 3.1: A Semi-Lagrangian Algorithm consists of determining the back trajectory T and interpolating the solution to x_o.

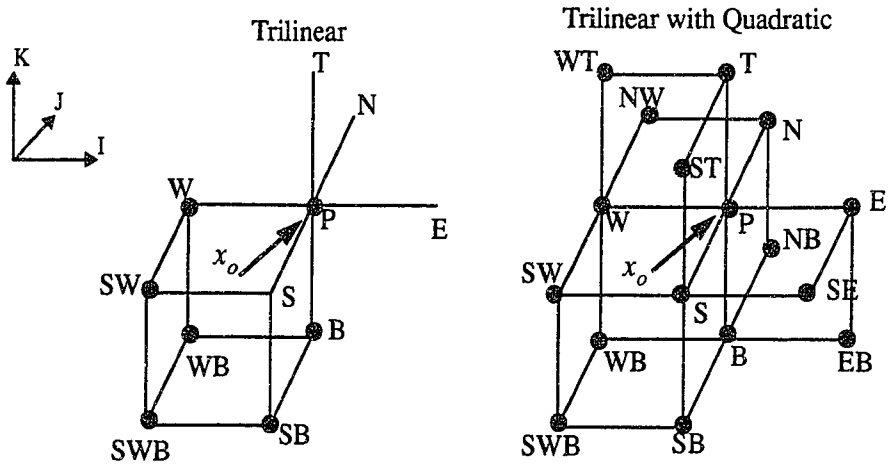

Figure 3.2: The trilinear basis for the case (u>0, v>0, w>0) uses the 8 points centered about the departure point. The higher order correction to the interpolation uses tangential and normal curvature terms to improve the interpolation order. Bold dots indicate points that are used in the stencil. The arrival point is P and N, S, E, W, T and B refer to the orientation of neighboring gridpoints around P.

This algorithm uses a combination of high order interpolation for accuracy in smooth regions of the flow and trilinear interpolation centered about the departure point for monotonicity near sharp gradients of ψ. Trilinear interpolation ensures that $\psi\left(\hat{x}_o, t^n\right)$ lies between the maximum and minimum ψ of the surrounding gridpoints. This is illustrated in the left-hand plot of Figure 3.2. We first consider constant velocity advection. In coordinates that have been scaled so the gridpoints lie at integer values of each coordinate, the departure point calculation takes the form

$$\hat{x}_o = \hat{x}_i - \hat{\nu} \tag{3.5}$$

where (ν_x, ν_y, ν_z) is the vector of Courant numbers $\vec{U}\Delta t/\Delta x$. The semi-Lagrangian algorithm based on trilinear interpolation takes the form

$$
\begin{aligned}
\psi^{n+1}(\xi_i, \eta_i, \zeta_i) \;&=\; \psi_p \\
&-\nu_x(\psi_p - \psi_w) - \nu_y(\psi_p - \psi_s) - \nu_z(\psi_p - \psi_b) \\
&+\nu_x\nu_y(\psi_p + \psi_{sw} - \psi_w - \psi_s) + \\
&+\nu_x\nu_z(\psi_p + \psi_{sb} - \psi_s - \psi_b) + \\
&+\nu_y\nu_z(\psi_p + \psi_{wb} - \psi_b - \psi_w) + \\
&-\nu_x\nu_y\nu_z(\psi_p + \psi_{sb} - \psi_s - \psi_b - \psi_w - \psi_{swb} + \psi_{sw} + \psi_{wb})
\end{aligned}
\tag{3.6}
$$

where the lack of superscripts indicates the use of values from t^n. This can be expressed in conservative form

$$
\psi_p^{n+1} \;=\; \psi_p - \left(F_e^{lo} - F_w^{lo}\right) - \left(F_n^{lo} - F_s^{lo}\right) - \left(F_t^{lo} - F_b^{lo}\right), \tag{3.7}
$$

The quantity F_e^{lo} represents the low-order (lo) flux through the east face of a grid cell centered at ψ_p. This flux can be written as

$$
F_e^{lo} \;=\; \nu_x\psi_p - \frac{\nu_x\nu_y}{2}(\psi_p - \psi_s) - \frac{\nu_x\nu_z}{2}(\psi_p - \psi_b) + \frac{\nu_x\nu_y\nu_z}{3}(\psi_p + \psi_{sb} - \psi_b - \psi_s)\;,
$$

This method is only first order accurate. It can be made second order accurate by including a normal slope correction to the flux.

$$
F_e^c \;=\; \frac{\nu_x}{2}(1 - \nu_x)\;(\psi_e - \psi_p) \tag{3.8}
$$

This correction is equivalent to adding a quadratic basis to the interpolation function in the coordinate directions. It was found to be unstable for three-dimensional flow directions, but can be stabilized by adding quadratic approximations to the third order tangential curvature terms in the error of the low order interpolation scheme. These tangential curvature terms are identical to the tangential curvature terms presented by Leonard (1993) for his UTOPIA scheme. The modified flux is

$$F_e^c = \frac{\nu_x}{2}(1-\nu_x)\,(\psi_e-\psi_p) - \frac{\nu_x\nu_y}{4}(1-\nu_y)\,(\psi_n-2\psi_p+\psi_s) - \frac{\nu_x\nu_z}{4}(1-\nu_z)\,(\psi_t-2\psi_p+\psi_b)\,. \tag{3.9}$$

This results in the interpolation stencil on the right-hand plot of Figure 3.2. Both the low and high order methods exhibit exact point to point transfer (i.e. perfect accuracy) for flows whose trajectories have departure points coincident with actual gridpoints.

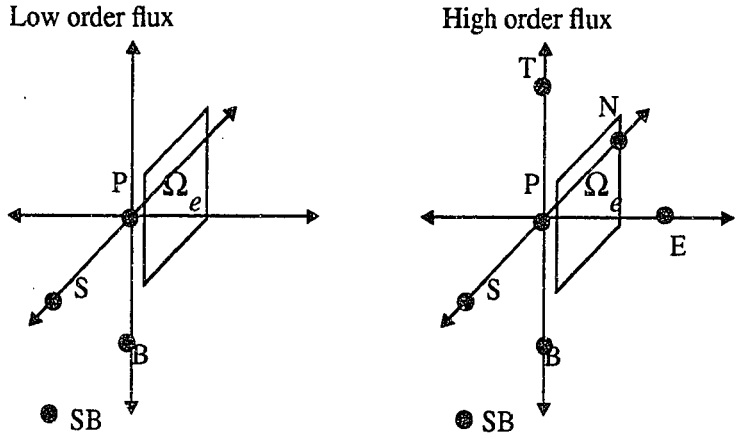

Figure 3.3: Illustration of the domain of dependence for the flux through a volume face. The rectangle indicates the volume face corresponding to the flux through the east face in Figure 3.2. Bold dots indicate points used in the stencil for the flux.

So far the method has been developed for a constant velocity field and a scalar field without sources. Next we generalize the method to handle variable velocity fields and sources while retaining second-order accuracy. It is again useful to take a semi-Lagrangian perspective on these generalizations. To apply this method for a variable velocity field, the constant Courant numbers (ν_x, ν_y, ν_z) in the above flux are replaced by the variable Courant numbers at the centers of the cell faces at time level n+1/2. The east side flux for the pseudo-incompressible flow with variable density becomes

$$F_e^{ho} = \rho_e v_{xe} \begin{pmatrix} \psi_p - \frac{v_{ye}}{2}(\psi_p - \psi_s) - \frac{v_{ze}}{2}(\psi_p - \psi_b) + \\ \frac{(1-v_{xe})}{2}(\psi_e - \psi_p) + \frac{v_{ye}v_{ze}}{3}(\psi_p + \psi_{sb} - \psi_b - \psi_s) - \\ \frac{v_{ye}}{4}(1-v_{ye})(\psi_n - 2\psi_p + \psi_s) - \frac{v_{ze}}{4}(1-v_{ze})(\psi_t - 2\psi_p + \psi_b) \end{pmatrix}. \quad (3.10)$$

The accuracy of this method for variable velocities and source terms can be analyzed by looking at the Taylor series expansion

$$\psi^{n+1} = \psi + \Delta t\left(\frac{\partial \psi}{\partial t}\right) + \frac{\Delta t^2}{2}\left(\frac{\partial^2 \psi}{\partial t^2}\right) + O(\Delta t^3). \quad (3.11)$$

We use the convective form to express the time derivatives as spatial ones:

$$\psi^{n+1} = \psi - \Delta t \begin{pmatrix} \left(u_j + \frac{\Delta t}{2}\frac{\partial u_j}{\partial t}\right)\frac{\partial \psi}{\partial x_j} - \frac{\Delta t}{2}u_j u_i \frac{\partial^2 \psi}{\partial x_i \partial x_j} - \frac{\Delta t}{2}\frac{\partial \psi}{\partial x_i}\frac{\partial u_i}{\partial x_j} \\ -R - \frac{\Delta t}{2}\frac{\partial R}{\partial t} + \frac{\Delta t}{2}u_j\frac{\partial R}{\partial x_j} \end{pmatrix}. \quad (3.12)$$

The upper line of terms arise from advection. The first term represents the advection of ψ by the velocity and its acceleration, the second term represents the curvature advection, and the third term represents the advection caused by spatial variation of the velocities. The lower line of terms come from the source term. A conservative form of the equations can be found by using the continuity equation and simplifying,

$$\psi^{n+1} = \psi - \frac{\Delta t}{\rho}\begin{pmatrix} \frac{\partial}{\partial x_j}\rho\left(u_j + \frac{\Delta t}{2}\frac{\partial u_j}{\partial t}\right)\psi - \frac{\Delta t}{2}\frac{\partial}{\partial x_j}\rho u_j u_i \frac{\partial \psi}{\partial x_i} \\ -\rho R - \rho\frac{\Delta t}{2}\frac{\partial R}{\partial t} + \frac{\Delta t}{2}\frac{\partial}{\partial x_i}(\rho u_j R) \end{pmatrix}. \quad (3.13)$$

This series is identical in form to the constant velocity case for time-independent flow. There are no spatial derivatives of the velocities inside the fluxes to differentiate between the two forms.

The time derivatives of the velocities disappear in both forms of the expansion, if one uses $u_j^{n+1/2}$ instead of u_j^n for the advective velocities. There is a semi-Lagrangian explanation why using velocities at $t^{n+1/2}$ and from the centers of the cell walls makes the algorithm second order accurate for temporally varying flows. If one examines the trajectories that parcels take in a variable flow, one can construct a first order approximation to the departure point of the trajectory. In tensor notation, this is expressed as

$$x_j^o = x_j^i - \Delta t u_j^i + O(\Delta t^2). \tag{3.14}$$

where the superscripts 'o' and 'i' denote the departure and arrival points. A second order approximation to the trajectory is found by expanding u_j^i in a Taylor series about $\left(x_j^i, t^{n+1/2}\right)$ and integrating the characteristic equation backwards in time using the first order trajectory to determine the parcel velocity.

$$\begin{aligned} x_j^o &= x_j^i - \int_0^{\Delta t} u_j(x_j(\tau), t^{n+1} - \tau)\, d\tau \\ &= x_j^i - \int_0^{\Delta t} \left(u_j^i + (-u_k^i \tau)\frac{\partial}{\partial x_k} u_j^i + \left(\frac{\Delta t}{2} - \tau\right)\frac{\partial u_j}{\partial t} \right) d\tau + O(\Delta t^3) \\ &= x_j^i - \Delta t u_j^i + \frac{\Delta t^2}{2} u_k^i \frac{\partial}{\partial x_k} u_j^i + O(\Delta t^3) \\ &= x_j^i - \Delta t (u_m)_j^i + O(\Delta t^3), \end{aligned} \tag{3.15}$$

where u_m is a second-order accurate approximation to the velocity at the midpoint of the trajectory at $t^{n+1/2}$. The quadratic curvature term $u_k \frac{\partial u_j}{\partial x_k}$ is an indicator of the curvature of trajectories in a fluid flow. Use of velocities from the centers of cell walls eliminates the need to treat this curvature term explicitly. The process by which the velocities are calculated at $t^{n+1/2}$ is derived in chapter 4.

The treatment of sources in our algorithm follows Smolarkiewicz and Pudykiewicz (1992). This treatment is a form of Strang Splitting (Strang 1968). We start from the Lagrangian form of the transport equation

$$\frac{D\psi}{Dt} = R, \tag{3.16}$$

which can be represented with Stokes' Theorem as the integral

$$\psi\left(\vec{x}, t^{n+1}\right) = \psi\left(\vec{x}_o, t^n\right) + \int_T R dt. \tag{3.17}$$

Here T is the parcel trajectory. A second order approximation to (3.17) is found using the trapezoidal rule,

$$\psi\left(\vec{x}, t^{n+1}\right) = \psi\left(\vec{x}_o, t^n\right) + \frac{\Delta t}{2}\left(R\left(\vec{x}_o, t^n\right) + R\left(\vec{x}, t^{n+1}\right)\right). \tag{3.18}$$

The trapezoidal rule is the ideal treatment of sources for its stability and accuracy properties, but if $R\left(\vec{x}, t^{n+1}\right)$ depends on ψ and hence is unknown, the Runge-Kutta analog is used,

$$\begin{aligned} \psi^*\left(\vec{x}, t^{n+1}\right) &= \psi\left(\vec{x}_o, t^n\right) + \Delta t R\left(\vec{x}_o, t^n, \psi^n\right), \\ \psi\left(\vec{x}, t^{n+1}\right) &= \psi\left(\vec{x}, t^n\right) + \frac{\Delta t}{2}\left(R\left(\vec{x}_o, t^n, \psi^n\right) + R\left(\vec{x}, t^{n+1}, \psi^*\right)\right). \end{aligned} \tag{3.19}$$

We denote the homogenous second order advection method as a linear operator A, with the property

$$\psi\left(\vec{x}_o, t^n\right) = A\left(\psi\left(\vec{x}, t^n\right), \vec{v}^{n+1/2}\right) + O(\Delta t^3). \tag{3.20}$$

Equation (3.18) can be approximated by advecting the quantity $\psi + (\Delta t R)/2$ with the homogenous advection algorithm and using the integration rule

$$\psi\left(\vec{x}, t^{n+1}\right) = A\left(\psi\left(\vec{x}, t^n\right) + \frac{\Delta t}{2} R\left(\vec{x}, t^n\right), \vec{v}^{n+1/2}\right) + \frac{\Delta t}{2} R\left(\vec{x}, t^{n+1}\right). \tag{3.21}$$

If R depends on $\psi\left(\vec{x}, t^{n+1}\right)$, the latter is replaced by $\psi^*\left(\vec{x}, t^{n+1}\right)$ as in (3.19). This procedure eliminates the need to compute $R\left(\vec{x}_o, t^n\right)$ and $\psi\left(\vec{x}_o, t^n\right)$ separately, and is discussed in further detail by Smolarkiewicz and Margolin (1993).

3.3 Eulerian Interpretation

This algorithm can also be derived in an Eulerian manner for the constant velocity case. By integrating the conservative form of the advection equation over a grid cell and using the divergence theorem, we obtain the exact form

$$\frac{\partial}{\partial t}\int_{\Omega} \rho\psi dV = -\int_{\partial\Omega} \rho u_n \psi dA. \tag{3.22}$$

If we integrate this equation with respect to time and interpret ψ as the average value of ψ in the grid volume, we obtain for the case of constant velocity and density,

$$\psi_p^{n+1} = \psi_p - \frac{1}{dxdydz}\int_0^{dt}\int_{\partial\Omega} u_n \psi dA. \tag{3.23}$$

The fluxes for this algorithm can be computed by looking at the left hand face of the volume and using trajectories to transform the integral into one that involves only ψ at time n.

$$\begin{aligned} F_e &= \frac{u}{dxdydz}\int_0^{dt}\int_{\partial\Omega_e} \psi\left(\frac{1}{2}, y, z, t^n + \tau\right) dA d\tau \\ &= \frac{u}{dxdydz}\int_0^{dt}\int_{\partial\Omega_e} \psi\left(\frac{1}{2} - u\tau, y - v\tau, z - w\tau, t^n\right) dA d\tau. \end{aligned} \tag{3.24}$$

The accuracy of evaluating this integral depends on the accuracy with which we represent the original solution at time level n. A second-order accurate flux can be found by assuming the solution varies linearly in each direction and bilinearly in the transverse directions.

$$\psi(\xi,\eta,\zeta) = \psi_p + \xi(\psi_e - \psi_p) + \eta(\psi_p - \psi_s) + \zeta(\psi_p - \psi_s) + \eta\zeta(\psi_p + \psi_{sb} - \psi_b - \psi_s). \tag{3.25}$$

The expression derived for F_e is the same as the semi-Lagrangian flux with a normal slope correction term.

$$F_e = \nu_x\psi_p - \frac{\nu_x\nu_y}{2}(\psi_p - \psi_s) - \frac{\nu_x\nu_z}{2}(\psi_p - \psi_b) + \frac{\nu_x\nu_y\nu_z}{2}(\psi_p + \psi_{sb} - \psi_b - \psi_s) + \frac{\nu_x}{2}(1 - \nu_x)(\psi_e - \psi_p). \tag{3.26}$$

The tangential curvature terms that stabilize this flux can be found by replacing the transverse gradient terms with a more accurate form. For example, the donor cell form of

$$-\frac{\nu_x}{2}(\nu_y(\psi_p - \psi_s)) \tag{3.27}$$

is replaced with the Lax-Wendroff form

$$-\frac{\nu_x}{2}\left(\nu_y(\psi_p - \psi_s) + \frac{\nu_y^2}{2}(\psi_n - 2\psi_p + \psi_s)\right). \tag{3.28}$$

The Eulerian interpretation of this algorithm is that it models the time evolution of a cell volume from the characteristics that impinge on the sides of the cell. This approach to deriving a forward in time method is similar to LeVeque (1993). LeVeque computes the fluxes through the sides of a cell by summing the contribution of a sequence of corrective waves that approximate the transverse propagation of an oblique grid.

3.4 Monotonicity

To preserve monotonicity, this algorithm breaks the flux down into low and high order fluxes,

$$F = F^{lo} + \Phi F^{c}. \tag{3.29}$$

Here F^{lo} represents the low order flux, F^{c} represents the corrective flux which is the difference between the high and low order fluxes, and Φ is the flux limiter. The low order flux can be written in a manner convenient for examining monotonicity:

$$\psi_p^{n+1} = a_1\psi_p + a_2\psi_s + a_3\psi_w + a_4\psi_b + a_5\psi_{sw} + a_6\psi_{wb} + a_7\psi_{sw} + a_8\psi_{wb} \tag{3.30}$$

where the weights a_i are positive and depend on the Courant numbers ν_i. These weights are given by trilinear interpolation and can be shown to satisfy a triangle inequality for both constant and variable velocity fields, such that

$$\sum_{k=1}^{8} a_k = 1 + \alpha h^2, \tag{3.31}$$

from which we deduce that

$$\left|\psi_p^{n+1}\right| \le (1 + \alpha h^2)\max(|\psi_i|), \tag{3.32}$$

where α is identically zero for the constant coefficient case and $h = \max(\Delta x, \Delta y, \Delta z)$. Mass continuity was assumed to eliminate the first-order term for variable velocity fields. This inequality (3.32) implies that numerical overshoots should be small, while including as many transverse terms as possible to minimize the filtering effect of the flux limiter.

The corrective flux is a normal slope correction that is modified by the third order tangential curvature terms. These curvature terms tend to diffusively stabilize the normal slope correction and contribute negligibly to numerical oscillations. The normal slope correction represents a quadratic dependence in the interpolation that can easily create oscillations. We can insure monotonicity in the combined algorithm by explicitly enforcing the condition

$$\psi_{\min} \le \psi_p^{n+1} \le \psi_{\max}, \tag{3.33}$$

where

$$\begin{aligned}\psi_{\min} &= \min(\psi_p^{n,*}, \psi_n^{n,*}, \psi_s^{n,*}, \psi_e^{n,*}, \psi_w^{n,*}, \psi_t^{n,*}, \psi_b^{n,*}) \\ \psi_{\max} &= \max(\psi_p^{n,*}, \psi_n^{n,*}, \psi_s^{n,*}, \psi_e^{n,*}, \psi_w^{n,*}, \psi_t^{n,*}, \psi_b^{n,*})\end{aligned} \tag{3.34}$$

$\psi_{\min}$ and $\psi_{\max}$ are the extrema of cells in the normal directions from the cell centers. ψ^* denotes values from the low-order method at time n+1. The low order solution values were included to prevent the limiter from being overly diffusive in regions where an extrema is actually being advected into a cell and not being spuriously created by dispersive error terms. (3.33) is the basis for the flux correction scheme of Zalesak (1979) and of Smolarkiewicz (1986). In the Zalesak scheme, the flux limiter for the east face of a cell is given by

$$\Phi_e = \begin{cases} \min\left(1, \dfrac{\psi_{\max} - \psi_j^*}{\sum F_{IN}^C}\right) & F_e^C < 0 \\ \min\left(1, \dfrac{\psi_j^* - \psi_{\min}}{\sum F_{OUT}^C}\right) & F_e^C > 0 \end{cases} \tag{3.35}$$

where F_{IN}^C and F_{OUT}^C are the net flux into and out of the cell. This flux limiter is a true multidimensional limiter in that it takes into account the total flux entering or leaving a cell. This is an improvement over using one-dimensional flux limiters applied in the different normal directions. In a multi-dimensional flow, one-dimensional limiting cannot account for multiple fluxes acting in concert to cause numerical oscillations.

3.5 Stability and Accuracy Analysis

For advection at a constant velocity, the stability of this algorithm can be analyzed using von Neumann analysis. We consider the advection of the field

$$\psi_{i,j,k}^n = \exp(\hat{i}\,(\theta_x i + \theta_y j + \theta_z k)) \qquad \hat{i} = \sqrt{-1} \tag{3.36}$$

to time n+1. The resulting field can be written in terms of a complex amplification factor $\hat{\psi}$ such that

$$\psi_{i,j,k}^{n+1} = \hat{\psi}(\nu_x, \nu_y, \nu_z, \theta_x, \theta_y, \theta_z) \exp(\hat{i}(\theta_x i + \theta_y j + \theta_z k)) \tag{3.37}$$

where $(\theta_x, \theta_y, \theta_z)$ are the wavenumbers allowed on the grid weighted by the grid spacing in each direction. The stability of any linear method for a given velocity can be found by looking at the maximum magnitude of $\hat{\psi}$ over all wavenumbers $(\theta_x, \theta_y, \theta_z)$.

$$S(\nu_x, \nu_y, \nu_z) = \max(|\hat{\psi}(\nu_x, \nu_y, \nu_z, \theta_x, \theta_y, \theta_z)|) \qquad -\pi \le \theta_x, \theta_y, \theta_z \le \pi \tag{3.38}$$

It is possible to derive a compact formula for the amplification factor of the low order method and thereby analyze its stability.

$$\begin{aligned} \hat{\psi} &= \left(1-\nu_x\left(1-e^{-i\theta_x}\right)\right)\left(1-\nu_y\left(1-e^{-i\theta_y}\right)\right)\left(1-\nu_z\left(1-e^{-i\theta_z}\right)\right) \\ S &= \max_{\theta_x}\left(1-\nu_x\left(1-e^{-i\theta_x}\right)\right)\max_{\theta_y}\left(1-\nu_y\left(1-e^{-i\theta_y}\right)\right)\max_{\theta_z}\left(1-\nu_z\left(1-e^{-i\theta_z}\right)\right). \end{aligned} \tag{3.39}$$

This indicates that this method is stable for the cube, $0 \le \nu_x, \nu_y, \nu_z \le 1$. $\hat{\psi}$ has a magnitude that drops off steeply for nonconstant wavenumbers. This is representative of the first order nature of this algorithm. In general, it is not possible to derive such a useful analytical formula for S even when $\hat{\psi}$ is known. However knowing $\hat{\psi}$, an estimate of the S can be found by numerically by iterating over the complete range of wavenumbers $(\theta_x, \theta_y, \theta_z)$ for a given Courant number. The resulting S can then be plotted as a function of Courant number. The method is stable inside the isosurface $S = 1$. For the high-order algorithm, this isosurface is shown in Figure 3.4. The stability region for this algorithm is also the cube, $0 \le \nu_x, \nu_y, \nu_z \le 1$. This is the optimal stability region for a nearest neighbor algorithm. The stability region of the Upwind, Leapfrog, and UTOPIA methods is shown in Figure 3.5. These methods have smaller stability regions of the form $0 \le \nu_x + \nu_y + \nu_z \le 1$. This is a substantial advantage of our method for use in multidimensional flow solvers.

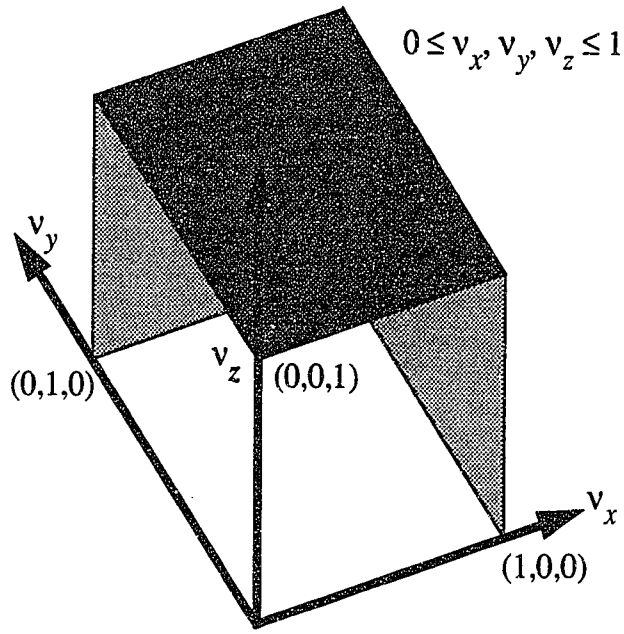

Figure 3.4: Figure of three dimensional stability regions. The solid curve is the isosurface where both high and low-order methods are marginally stable. The coordinate axes are Courant number in *x*, *y* and *z*.

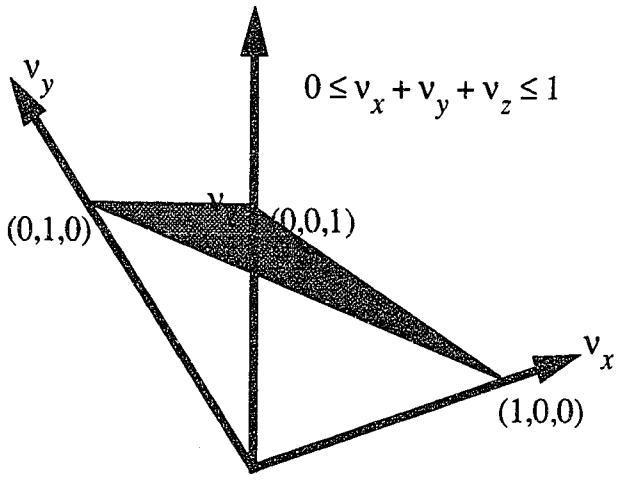

Figure 3.5: Figure of three dimensional stability region for the UTOPIA, leapfrog, and upwind algorithms.

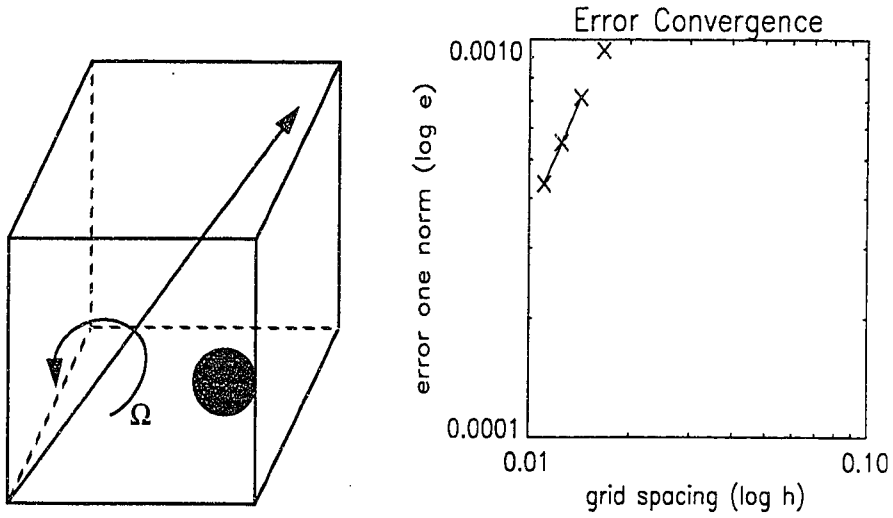

Figure 3.6: Three dimensional error convergence test. Solid body rotation of a three dimensional spherical Gaussian distribution about the vector (1,1,1). The right hand plot shows the second order convergence results for a half of a rotation on resolution varying from 60 to 90 cubed gridpoints. The line represents second order convergence for the 70 to 90 cubed runs.

The error convergence of the high-order algorithm was tested to confirm the order of accuracy for a spatially varying three-dimensional flow. A spherical Gaussian distribution was advected with solid body rotation about the vector $(1, 1, 1)$ for a half rotation and the 1-norm of the error was calculated for several grid spacings. The log-log plot in Figure 3.6 shows that the slope of the error curve is second order. A least-squares fit of the data reveals a slope of 1.98.

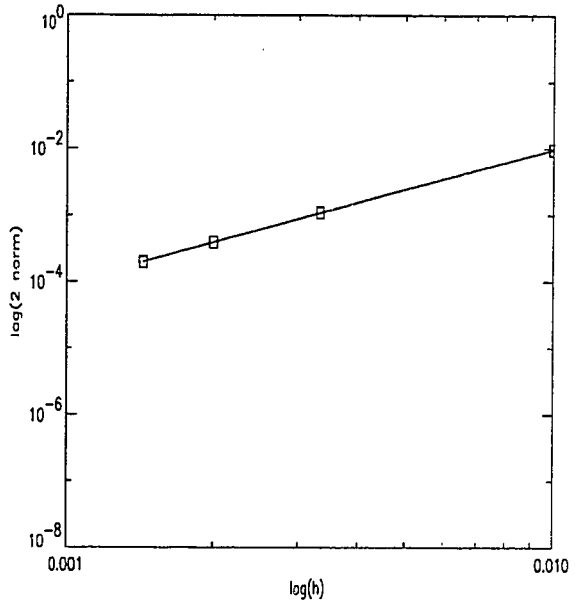

Figure 3.7: Error convergence results for an advection test involving a variable in time and space source term. The line represents second order convergence.

A numerical test of the accuracy of the treatment of sources is given by the model problem.

$$\begin{aligned} u_t + u_x &= \sin(2\pi t)\sin(2\pi x) \qquad 0 \le x \le 1 \qquad 0 \le t \\ u(x,0) &= \exp(-64(x-1/2)^2) \\ u(0,t) &= u(1,t) \end{aligned} \tag{3.40}$$

The log-log plot in Figure 3.7 shows a least-square fit of slope 1.99 for the error convergence.

3.6 Comparison with other advection methods

Many previous meteorological models have been formulated about methods using multiple timelevels such as the leapfrog algorithm. This algorithm uses central differencing in both space and time,

$$\psi^{n+1} = \psi^{n-1} - \frac{2\Delta t}{\rho}(\nabla\cdot\rho\vec{U}\psi)^n. \tag{3.41}$$

It is a simple single-step method that is second-order accurate and non-dissipative. It requires storing one extra time level and has numerical dispersive errors due to the central differencing. Leapfrog algorithms require special treatment to eliminate the computational mode that arises from the eventual decoupling of odd and even timelevels. One technique to eliminate this computational mode is to perform a corrective trapezoidal step

$$\begin{aligned} \psi^{*} &= \psi^{n-1} - \frac{2\Delta t}{\rho}(\nabla\cdot\rho\vec{U}\psi)^n, \\ \psi^{n+1/2} &= (\psi^{*} + \psi^{n})/2, \\ \psi^{n+1} &= \psi^{n} - \frac{2\Delta t}{\rho}(\nabla\cdot\rho\vec{U}\psi)^{n+1/2}. \end{aligned} \tag{3.42}$$

This retains the second-order accuracy, but it doubles the amount of work required. Another technique often used is temporal filtering (Robert 1966; Asselin 1972).

$$\begin{aligned} \psi^{n+1} &= \bar{\psi}^{n-1} - \frac{2\Delta t}{\rho}(\nabla\cdot\rho\vec{U}\psi)^n, \\ \bar{\psi}^{n} &= \psi^{n} + \frac{\varepsilon}{2}(\psi^{n+1} - 2\psi^{n} + \bar{\psi}^{n-1}), \end{aligned} \tag{3.43}$$

where the barred terms represent filtered variables and ε is a filtering constant. This technique avoids increasing the amount of work, but the temporal filtering makes the overall method only first order accurate. The use of central differencing to evaluate the convective terms creates spatial decoupling as well. It is possible to use flux-correction for monotonicity as well. However, the flux limiter is often invoked in smooth regions of the flow as gridpoints decouple. This reduces the accuracy of the method to first order in these regions and negates the advantage of being non-dissipative.

We compared our algorithm (hereafter modified Leonard algorithm) with flux-correction to the flux-corrected leapfrog-trapezoidal algorithm of Zalesak (1979) and a temporally filtered flux-corrected leapfrog algorithm. In a test similar to the test in Figure 3.6, a three-dimensional sphere of concentration 1 with a rectangular wedge cut out of it was

advected in solid body rotation about the vector (1,1,1) for a complete revolution over a range of resolutions. After one complete revolution of the slotted sphere, we computed the one-norm of the error between the true solution and the numerical solution as an estimate of the accuracy of the algorithm. This is a difficult test of an algorithms ability to track sharp gradients without introducing dispersion or dissipation (Figure 3.8). This test is a three-dimensional analog to the familiar slotted disk test (Zalesak 1979; Smolarkiewicz 1986). The error convergence of these methods is plotted in Figure 3.9. The leapfrog-trapezoidal algorithm with flux correction is remarkably monotonic. The modified Leonard method was shown to be more accurate than the leapfrog methods.

We compared our algorithm with another forward-in-time algorithm (Smolarkiewicz 1984). This algorithm makes relies on the monotonicity-preserving property of the upwind algorithm. This is a predictor-corrector method that uses the upwind method as a predictor and the upwind method with specially formulated anti-diffusive velocities as the corrector. The anti-diffusive velocities depend on the solution making this algorithm nonlinear. This has the disadvantage that it cannot be used simply for advected quantities such as velocities that are symmetric about zero. This algorithm is 15% faster than the modified Leonard algorithm due to the extra upwinding that occurs in our algorithm. The error convergence test in Figure 3.10 shows that the modified Leonard algorithm is about 20% more accurate. The modified Leonard algorithm is stable for a much larger range of Courant numbers. The error convergence for the modified Leonard algorithm with twice the timestep is presented in Figure 3.10. The error curve for the doubled timestep coincides closely with the lower timestep curve. The doubled timestep results in Courant numbers for which all of the other algorithms are unstable.

The simulation of turbulent flow requires advection methods that do not introduce a grid bias into their flows. A three-dimensional test problem was set up as an analog to the two dimensional anisotropy test of Leonard (1993). This test involves advecting a cosine hump $\cos^2(\pi r/2)$ with a uniform velocity of (1,1,1) and triply periodic boundary conditions (Figure 3.11). For a smooth initial profile, the different methods produced anisotro-

pic results. The anisotropy of a method was found by looking at a contour plot of the solution in a vertically oriented plane containing the origin and the velocity vector. Anisotropy appears as the deviation of the contours from perfect circles. Figure 3.12 shows the results for the unmodified leapfrog algorithm with no flux correction and temporal filters, Figure 3.13 is for the non-flux corrected Smolarkiewicz scheme, and Figure 3.14 the result for the modified Leonard algorithm. For all three tests, the Courant numbers had magnitude of 0.2 in each direction. For constant velocities, the modified Leonard algorithm reduces identically to the semi-Lagrangian algorithm presented earlier in Section 3.2. This test clearly illustrates the ability of a semi-Lagrangian based algorithm to propagate information accurately transverse to the grid. The decrease in anisotropy of the modified Leonard algorithm is due to the ability of a semi-Lagrangian algorithm to track characteristics that are not aligned with the grid.

We have shown that using the semi-Lagrangian perspective, while maintaining Eulerian conservation that we can make an advection algorithm that has optimal stability and competitive accuracy. Its computational efficiency is comparable to that of other forward-in-time algorithms.

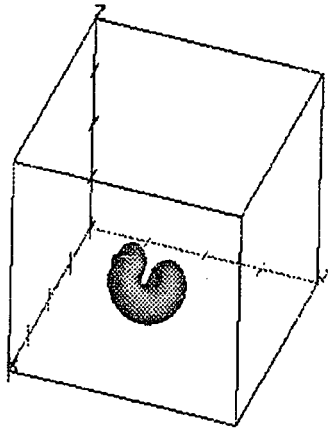

Figure 3.8: Illustration of final state of the advection test. The isosurface shown is the spare with the missing rectangular wedge after one complete revolution about the axis (1,1,1).

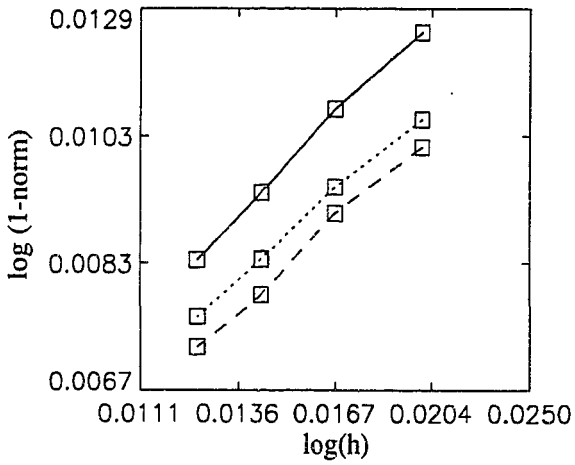

Figure 3.9: Comparison of the flux-corrected versions of trapezoidal-leapfrog (dotted line), Robert-Asselin filtered leapfrog (dotted) and modified Leonard algorithm (dashed).

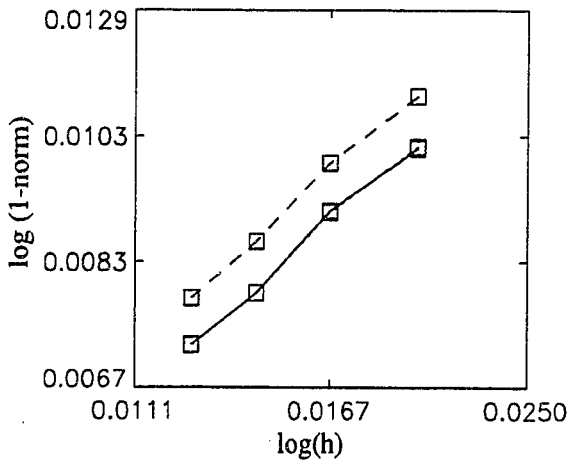

Figure 3.10: Error convergence results for Smolarkiewicz (dashed) and modified Leonard algorithm (solid). The doubled timestep curve is nearly coincident with the smaller timestep curve.

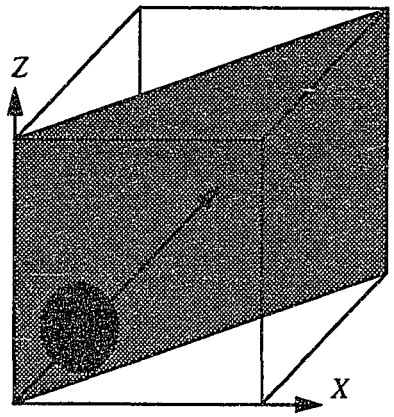

Figure 3.11: Illustration of the Anisotropy test. A cosine distribution is advected along (1,1,1). The *y* direction is into the page.

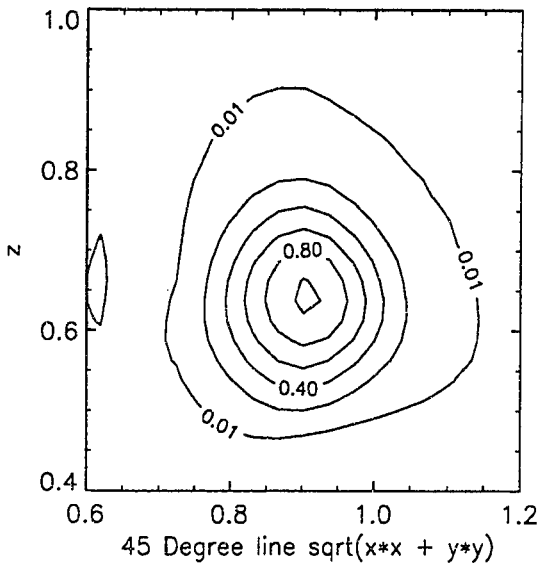

Figure 3.12: Isotropy test for the unmodified leapfrog scheme.

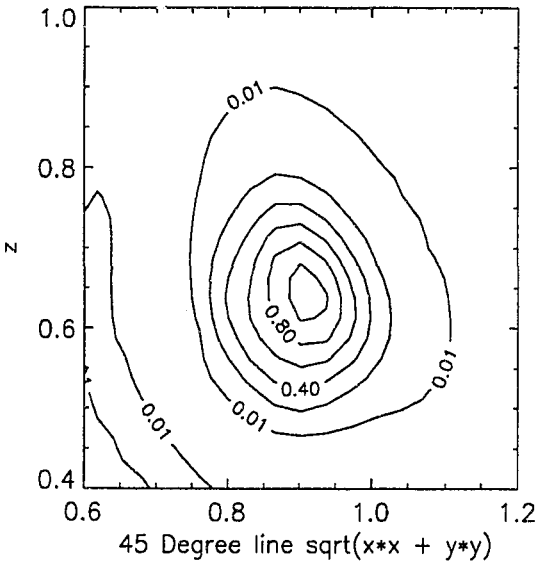

Figure 3.13: Isotropy test for the Smolarkiewicz scheme with no flux correction.

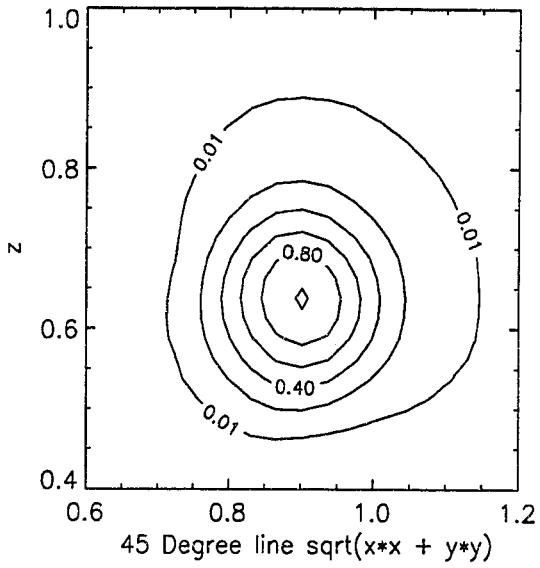

Figure 3.14: Isotropy test for the modified Leonard scheme with no flux correction.

4 Numerical Solution of the Anelastic Equations

4.1 Numerical Algorithm

Atmospheric convection is a complex process in which a pseudo-incompressible fluid is involved in such physical processes as rain, radiation and condensation. A chief difficulty in numerically simulating this phenomenon is providing a general framework by which physical processes are implemented in an accurate manner. The popularity of leapfrog methods relies on the fact that they provide a conceptually simple framework to handle arbitrary source terms and variable velocities with second order accuracy, despite their problems with stability and monotonicity. Forward-in-time methods do not suffer from these problems, but accurate ways of including variable source terms and velocities have only recently been developed. Efforts to combine the advantages of forward-in-time advection with the generality of leapfrog schemes have resulted in hybrid schemes (Beheng 1994; Skamarock and Klemp 1992). Leapfrog is used to advance the dynamics and provide advective velocities that are used by a forward-in-time advection scheme to advance advected scalars. The hybrid schemes realize the benefits of a forward-in-time scheme for at least the scalars, but their stability and accuracy are still limited by the time-filtering needed to stabilize the leapfrog scheme.

Our advection method provides a general second-order accurate framework for advection with arbitrary source terms and pseudo-incompressible velocities. Its stability and accuracy compare favorably with other advection methods. This chapter focuses on how the method can be used as part of a flow solver for the anelastic equations. The chief complication is the computation of the pressure such that velocities at $t^{n+1/2}$ and t^{n+1} preserve mass continuity within each grid volume. Our flow solver has none of the disadvantages of a hybrid or pure leapfrog scheme

For simplicity, we present an algorithm for integrating the inviscid anelastic equations for a dry environment. These equations demonstrate with the most clarity our treatment of

the dynamics and can easily be extended to handle moisture, diabatic effects, and subgrid scale turbulence. The anelastic equations for a dry environment (see Chapter 2) can be summarized as

$$\frac{du_i}{\partial t} + \frac{1}{\rho}\left(\frac{\partial}{\partial x_j}\rho u_j u_i\right) = -\frac{\partial \phi}{\partial x_i} + \delta_{i3} B\left(\theta^*\right),$$

$$\frac{\partial \rho u_i}{\partial x_i} = 0, \tag{4.1}$$

$$\frac{\partial \theta^*}{\partial t} + \frac{1}{\rho}\left(\frac{\partial}{\partial x_j}\rho u_j \theta^*\right) = 0,$$

where the quantity $\phi = C_{pd}\theta_o \pi''$ is often referred to as the Exner function and $\rho = \bar{\rho}(z)$ is the density associated with an isentropic base state. These equations have been nondimensionalized with the following scalings:

$$\begin{aligned}
\vec{U} &= \frac{\vec{U}^*\,(\mathrm{m/s})}{\langle U \rangle} \\
\vec{x} &= \frac{\vec{x}^*\,(\mathrm{m})}{H} \\
\rho &= \frac{\rho^*\,(\mathrm{kg/m^3})}{\langle \rho \rangle} \\
P &= \frac{P^*\,(\mathrm{mbar})}{\langle \rho \rangle \langle U \rangle^2} \\
t &= \frac{t^*\,(\mathrm{s})}{\tau}
\end{aligned} \tag{4.2}$$

Here asterisks indicate dimensional quantities and the advective timescale $\tau = H/\langle U \rangle$ has been used. The gravitational acceleration g has been replaced by a dimensionless number $\nu_B = gH\langle U \rangle^{-2}$ in the buoyancy $B = \nu_b(\theta^* - \theta^*_{env})$. Nondimensionalization of the equations helps to ensure that quantities integrated by the model are *O(1)*. For instance, the pressure scaling $\langle P \rangle$ is equal to 1 millibar for the scales $\langle \rho \rangle = 1\mathrm{kg/m}^3$ and $\langle U \rangle = 10\mathrm{m/s}$, a typical size for pressure perturbations in atmospheric convection.

The chief difficulty in integrating the dry anelastic equations is maintaining the anelastic continuity equation. Symbolically, we can step forward the system (4.1) using the advection algorithm from chapter 3:

$$u_i^{n+1} = A\left(u_i^n + \frac{\Delta t}{2}\left(\delta_{i3}B^n - \frac{\partial \phi^n}{\partial x_i} \right), \hat{\nu}^{n+1/2} \right) + \frac{\Delta t}{2}\left(\delta_{i3}B^{n+1} - \frac{\partial \phi^{n+1}}{\partial x_i} \right),$$
$$\theta^{*n+1} = A\left(\theta^{*n}, \hat{\nu}^{n+1/2} \right), \tag{4.3}$$

where $A(\psi, \hat{\nu})$ is the advection operator defined in Equation (3.20) with second-order accuracy and stability. This requires the evaluation of velocities $\hat{\nu}^{n+1/2}$ and pressure ϕ^{n+1}. We insist that the velocities computed at n+1/2 and n+1 both satisfy the discretized anelastic continuity equation. This requires two pressure solves per timestep that are described in Section (4.2). The full solution of (4.3) proceeds as follows:

1. Calculate the momentum forcing at time n. If it is the start of the simulation or the start of a new grid, then a pressure solve is necessary.
2. Use the momentum equation to compute a first order accurate midpoint velocity u_i^m at time level n+1/2. The Courant numbers $\hat{\nu}^{n+1/4}$ and buoyancy $B^{n+1/2}$ have been replaced by their values at level n. However, for stability it is necessary to exactly preserve discrete mass continuity, so we require a pressure solve for ϕ^m to insure that ρu_i^m is nondivergent. The pressure solver is discussed in Section 4.2.

$$u_i^m = A\left(u_i^n + \frac{\Delta t}{4}\left(\delta_{i3}B^n - \frac{\partial \phi^n}{\partial x_i} \right), \hat{\nu}^n \right) + \frac{\Delta t}{4}\left(\delta_{i3}B^n - \frac{\partial \phi^m}{\partial x_i} \right). \tag{4.4}$$

3. Compute the advection of θ^* using the midpoint velocities, where $\hat{\nu}^m$ is the vector of Courant numbers associated with u_i^m

$$\theta^{*n+1} = A\left(\theta^{*n}, \hat{\nu}^m \right). \tag{4.5}$$

4. Advance the velocities using the modified trapezoidal rule. This requires a pressure solve for ϕ^{n+1} to ensure that ρu_i^{n+1} is nondivergent.

$$u_i^{n+1} = A\left(u_i^n + \frac{\Delta t}{2}\left(\delta_{i3}B^n - \frac{\partial \phi^n}{\partial x_i} \right), \hat{\nu}^m \right) + \frac{\Delta t}{2}\left(\delta_{i3}B^{n+1} - \frac{\partial \phi^{n+1}}{\partial x_i} \right). \tag{4.6}$$

The features of this algorithm are similar to the algorithms of Bell and Marcus (1989) and Smolarkiewicz and Margolin (1993). These schemes involve reusing the momentum

equation to extrapolate velocities in time to n+1/2 and requiring them to be discretely non-divergent for advection. The case with pressure and buoyancy as the only source terms was presented, since it illustrates the trapezoidal manner in which the momentum forcing terms are handled. For other source terms, the Runge-Kutta method of Section (3.2) is used.

4.2 Discrete Mass Continuity

The pressure in an incompressible flow adjusts the velocities so as to preserve mass continuity. In our algorithm (4.3), the velocities are stepped forward using an equation of the form

$$\vec{U}^{n+1} = \vec{U}^{*} - \frac{\Delta t}{2}\nabla\phi^{n+1} \tag{4.7}$$

where

$$u_i^{*} = A\left(u_i^{n} + \frac{\Delta t}{2}\left(\delta_{i3}B^{n} - \frac{\partial \phi^{n}}{\partial x_i} \right), \vec{v}^{m} \right) + \frac{\Delta t}{2}\delta_{i3}B^{n+1} \tag{4.8}$$

is the known divergent velocity that is the result of adding in all the advective and source terms for the nth timestep. The anelastic continuity equation then implies that

$$\frac{\Delta t}{2}\nabla\cdot\rho\nabla\phi = \nabla\cdot\rho\vec{U}^{*} = D^{*}, \tag{4.9}$$

This elliptic equation for the pressure is analogous to the Poisson equation one gets by taking the divergence of the momentum equations to remove the time tendency. The boundary conditions for the pressure equation are determined by the normal velocity at the boundary or by periodicity. In the case of periodic boundaries, the boundary conditions are

$$\phi(0, y, z) = \phi(L_x, y, z) \qquad \phi(x, 0, z) = \phi(x, L_y, z) \tag{4.10}$$

and for cases where the normal velocity u_n to a boundary is known, we have a Neumann boundary condition on ϕ:

$$u_n = u_n^* - \frac{\Delta t \partial \phi}{2 \partial n} \tag{4.11}$$

For rigid walls, $u_n = 0$. For both types of boundary conditions, these equations have a solvability requirement that arises from Gauss's theorem,

$$\int_{\partial\Omega} \rho u_n dA = 0 \tag{4.12}$$

No net mass can flow through the boundary $\partial\Omega$ of the domain Ω if all boundaries are rigid or periodic, this requirement is satisfied automatically.

Modern finite-difference algorithms are generally formulated in "conservation form" whenever possible, since a discrete analogue to a conservation law obeyed by a fluid helps to insure that even in regions of rapid change in flow properties, the numerical scheme will not produce highly unphysical results. Maintaining a discrete analogue to mass conservation was a strong priority in the design of our solver. Our flow solver uses the MAC (Marker and Cell) formulation of Harlow and Welch (1965), where the velocities are staggered one half gridpoint in the normal direction as shown in Figure 4.1. This formulation is also known as the Arakawa C-grid (Arakawa 1966) in the meteorological literature. This formulation was chosen since it offers a simple approach to mass conservation. In the compass point notation of Figure 4.1, the MAC formulation of the divergence and pressure force are:

$$D(\vec{U}) = \begin{bmatrix} \frac{\rho_p}{\Delta x}(u_e - u_w) + \\ \frac{\rho_p}{\Delta y}(v_n - v_s) + \\ \frac{\rho_t}{\Delta z} w_t - \frac{\rho_b}{\Delta z} w_b \end{bmatrix}, \tag{4.13}$$

and

$$(\phi_x)_e = \frac{(\phi_e - \phi_p)}{\Delta x}. \tag{4.14}$$

This results in the following stencil for the interior pressure.

$$\frac{\Delta t}{2}\begin{bmatrix} \frac{\rho_p}{\Delta x^2}(\phi_e - 2\phi_p + \phi_w) + \\ \frac{\rho_p}{\Delta y^2}(\phi_n - 2\phi_p + \phi_s) + \\ \frac{\rho_t}{\Delta z^2}(\phi_t - \phi_p) - \frac{\rho_b}{\Delta z^2}(\phi_p - \phi_b) \end{bmatrix} = D^*. \tag{4.15}$$

This formulation pairs the pressure force and the divergence to provide a discrete conservation of mass. The diagnostic equation for the pressure is the standard 7 point Laplacian and can be solved by fast direct Poisson solvers. For the following boundary conditions, this stencil provides a symmetric positive definite system of equations that is well-conditioned for all wavenumbers.

For periodic boundary conditions (4.10), one can assume periodicity in the pressure:

$$\phi_{0,j,k} = \phi_{nx,j,k}\ , \qquad \phi_{nx+1,j,k} = \phi_{1,j,k}\ . \tag{4.16}$$

For a boundary condition where the normal flow is specified, the boundary condition (4.11) becomes

$$u_{0,j,k} = u^*_{0,j,k} - \frac{(\phi_{1,j,k} - \phi_{0,j,k})}{\Delta x}. \tag{4.17}$$

Here, grid extension has been used to represent boundary conditions. These discrete boundary conditions are analogous to those of the continuous case. They result in the discrete solvability condition

$$\int_{\partial\Omega} \rho u_n dA \approx \sum_{i \in \partial\Omega} \rho_i u_{ni} dA_i = 0\ . \tag{4.18}$$

This formulation of the boundaries and of the interior pressure ensures that the discrete analog of mass conservation is maintained:

$$\int_{\Omega} \rho dV \approx \sum_{i \in \Omega} \rho_i dV_i = \text{Const.} \tag{4.19}$$

Recent developments in methods for incompressible fluids have suggested the use of non-staggered grids for the velocities. Though the formulation of mass conservation on a staggered grid is consistent for Cartesian coordinates, it does not generalize easily to a curvilinear coordinate system. Staggered grids complicate the treatment of advection considerably by requiring interpolation of advective velocities for velocity advection. Chorin (1969) used the Hodge decomposition, the fact that any velocity can be decomposed into a divergence free velocity plus the gradient of a scalar, to construct a non-staggered algorithm. By interpreting ϕ as this scalar, one can think of (4.7) as projecting $\vec{U}^*$ onto the space of divergence free vector fields. Methods based on this idea are called projection methods. The staggered velocity formulation given here is an example of such a projection. It is also possible to define projection methods for unstaggered grids (Chorin 1969; Marcus and Bell 1989). A general description of the numerical properties of unstaggered grids is given by Shih (1989).

Non-staggered treatments of mass conservation have to be carefully specified for their solutions to be described at all wavenumbers. Often the pressure force and divergence are evaluated by central differencing. This causes gridpoints to decouple in a manner similar in principle to a leapfrog advection algorithm. This tends to complicate the discretization and solution process for pressures associated with non-staggered velocities. Local decoupling is a strong reason for the continued popularity of methods using velocities on staggered grids. A promising approach taken by some modelers is to enforce discrete conservation of mass only on advective velocities which have been interpolated onto a staggered grid at half time levels (Almgren 1993; Zhang 1994). The unstaggered velocities are allowed to be only approximately divergence free. This eliminates the need for staggered advection equations, while maintaining a well conditioned pressure equation.

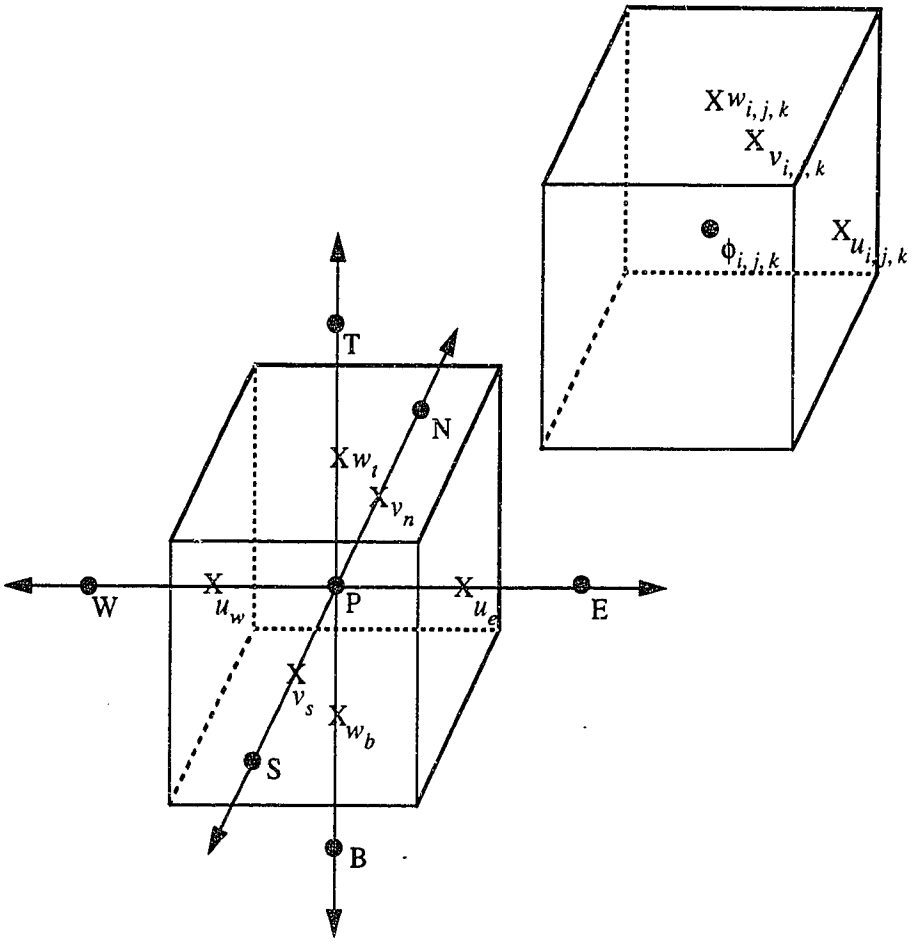

Figure 4.1: Illustration of grid staggering and grid indexing for the staggered grid used in this model. X's indicate velocity positions and dots indicate pressure gridpoints. The cell indices run from (1,1,1) to (NX,NY,NZ).

4.3 Numerical pressure solver

This model uses a fast direct method using FFT's to solve the pressure equation. The equations for a horizontal line in the x direction can be written as

$$\left[\begin{array}{l} \frac{\rho_k}{\Delta y^2}(\vec{\phi}_{j+1,k} - 2\vec{\phi}_{j,k} + \vec{\phi}_{j-1,k}) + \\ \quad \frac{\rho_{k+\frac{1}{2}}}{\Delta z^2}(\vec{\phi}_{j,k+1} - \vec{\phi}_{j,k}) - \\ \quad \frac{\rho_{k-\frac{1}{2}}}{\Delta z^2}(\vec{\phi}_{j,k} - \vec{\phi}_{j,k-1}) \end{array}\right] + \frac{\rho_k}{\Delta x^2} A\vec{\phi}_{j,k} = \vec{D^*}_{j,k} \tag{4.20}$$

where the unknowns have been rearranged into the vector $\vec{\phi}_{j,k} = \left[\phi_{1,j,k} \cdots \phi_{nz,j,k}\right]^T$.

For the case of periodic boundary conditions in the x direction, the matrix A represents the periodic structure

$$A = \begin{bmatrix} -2 & 1 & & & 1 \\ \cdots & \cdots & \cdots & & \\ & 1 & -2 & 1 & \\ & & \cdots & \cdots & \cdots \\ 1 & & & 1 & -2 \end{bmatrix} \tag{4.21}$$

The eigenvalues and eigenvectors of A are given by

$$\lambda_m = -4\sin^2\left(\frac{\pi m}{nx}\right) \qquad X_{mj} = e^{\frac{2\pi imj}{nx}} = \omega_{nx}^{mj} \tag{4.22}$$

where ω_{nx}^{mj} is the nxth root of unity and the indices m and j range from 0 to nx-1. When the equation is transformed into eigenspace using FFT's, it becomes decoupled in the x direction. Similarly the equations can be decoupled in the y direction. This process is done using software from the NAG library. The end result of this process is a decoupled set of equations, except in the vertical, which is represented by the tridiagonal matrix

$$\begin{bmatrix} \left(-\rho_{\frac{1}{2}}+\rho_1\alpha_{mn}\right) & \rho_{\frac{1}{2}} & & & \\ \cdots & \cdots & \cdots & & \\ & \rho_{k-\frac{1}{2}} & \left(-2\left(\rho_{k-\frac{1}{2}}+\rho_{k+\frac{1}{2}}\right)+\rho_k\alpha_{mn}\right) & \rho_{k+\frac{1}{2}} & \\ & & \cdots & \cdots & \cdots \\ & & & \rho_{nz-\frac{1}{2}} & \left(-\rho_{nz-\frac{1}{2}}+\rho_{nz}\alpha_{mn}\right) \end{bmatrix} \tag{4.23}$$

where the coefficient α_{mn} is a combination of the horizontal eigenvalues and aspect ratios of the grids, given by

$$\alpha_{mn} = \frac{\Delta z^2}{\Delta x^2}\lambda_m + \frac{\Delta z^2}{\Delta y^2}\lambda_n . \tag{4.24}$$

After the tridiagonal system is solved, the solution can be found by transforming back first in y then in x.

For the case where the velocity normal to a boundary is specified, the matrix A has the structure

$$A = \begin{vmatrix} -1 & 1 & & & \\ \cdots & \cdots & \cdots & & \\ & 1 & -2 & 1 & \\ & & \cdots & \cdots & \cdots \\ & & & 1 & -1 \end{vmatrix} \tag{4.25}$$

The eigenvectors and eigenvalues of this matrix are

$$\lambda_m = -4\sin^2\left(\frac{\pi m}{2nx}\right) \qquad X_{mj} = \cos\left(\frac{\pi}{nx}\left(m+\frac{1}{2}\right)\left(j+\frac{1}{2}\right)\right) \tag{4.26}$$

where m and j are indices that run from 0 to $nx-1$. This transformation is known as the quarter wavelength cosine transform. The FFT packing procedure for performing this transformation is discussed in Wilhelmson and Erickson (1977). As before, one decouples the equations by transforming and then solves a system of tridiagonal equations where the

term α_{mn} is altered by the new eigenvalues. The final solution is found by transforming back.

For both of these choices of boundary conditions, we treat the null space of the Poisson equation by noting that it is confined to the tridiagonal equation associated with the constant modes in the x and y directions. Special care is taken in the tridiagonal solver to solve for the orthogonal complement of this mode.

4.4 Model Validation

The anelastic system of equations conserve an energy quantity that presents us with an opportunity for testing the nonlinear form of the numerics. This energy is found by taking the dot product of the momentum equation with the velocity to obtain a mechanical energy equation.

$$\bar{\rho}\frac{d}{dt}(|\vec{U}|^2/2) = -\nabla\bullet P''\vec{U} + \bar{\rho}gw\left(\frac{\theta''}{\theta_o}\right) \tag{4.27}$$

where the anelastic continuity equation and base state have been used to simplify this expression. The mechanical energy is related to the internal energy through the relationship

$$\frac{d}{dt}\left(\frac{\theta''}{\theta_o}\right) + \frac{wN^2}{g} = 0 \tag{4.28}$$

where N^2/g is the stratification of the environmental perturbation θ'. For the case of uniform stratification, the internal energy equation becomes

$$\bar{\rho}\frac{d}{dt}\left(g\frac{(\theta''/\theta_o)^2}{2N^2}\right) = -\bar{\rho}gw\left(\frac{\theta''}{\theta_o}\right) \tag{4.29}$$

Hence we can define an energy,

$$E = \frac{|\vec{U}|^2}{2} + \frac{(g\theta''/N\theta_o)^2}{2} \tag{4.30}$$

whose only source term is the pressure work term.

$$\bar{\rho}\frac{DE}{Dt} = -\nabla \bullet P'' \vec{U} \tag{4.31}$$

This is the energy associated with gravity waves and the pressure work term is the energy flux of gravity waves (Durran 1989; Lighthill 1986). For the case of nonuniform stratification, an analogous energy quantity, $E = |\vec{U}|^2/2 - gz\theta^*$ can be derived (Ogura and Phillips 1962). When integrated over a domain with rigid or periodic boundaries, E is conserved.

A bubble collapse experiment was done to test the energy conservation properties of the model. This experiment is similar to those done by Orlanski (1976) and Clark and Farley (1984). The model was initialized with a bubble of radius 300 meters and potential temperature perturbation of 0.5 K in a still environment of constant static stability $N = .01$ s^{-1}. This initialization leads to the nonlinear excitation of gravity waves. The average energy for thirty minutes of simulation is plotted in Figure 4.2.

The total energy is initially the potential energy of the bubble. This energy is converted into kinetic energy as it rises to its level of neutral buoyancy at which point the oscillations of kinetic and potential energy are the result of gravity waves being transmitted throughout the domain. This gravity wave energy is trapped in the domain by the periodic and rigid lid boundary conditions.

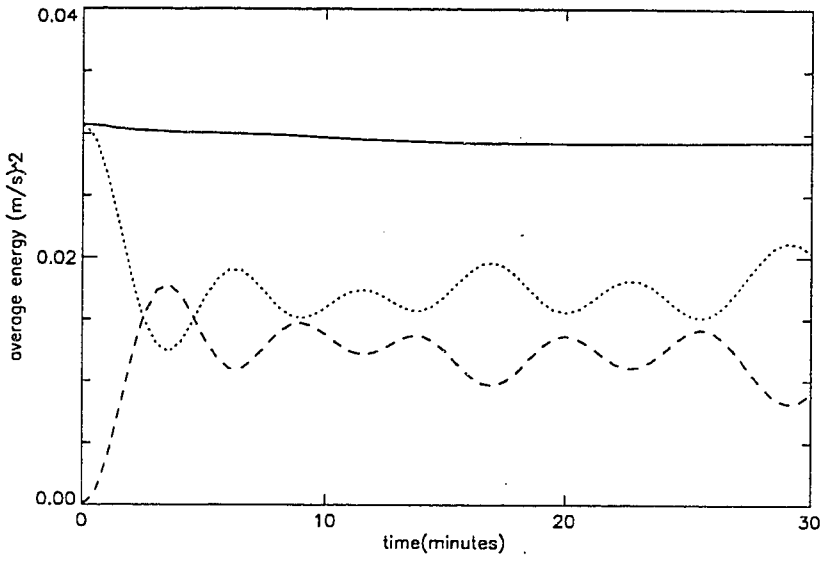

Figure 4.2: Conservation of total energy (solid line). The dotted line is potential energy and the dashed line is kinetic energy.

Linearized solutions of the equations for nonhydrostatic flow provide a qualitative tool for validating the numerics of a model (Clark 1977; Skamarock and Klemp 1994). We compared our algorithm to a spectral solution of the linearized equations for a localized heat source. This solution was adapted from one given by Nicholls (1991) in which the flow was assumed to be hydrostatic. In the presence of a linearly stratified environmental sounding and a uniform density profile, these equations take the form

$$\begin{aligned} \frac{\partial}{\partial t}\rho u + \frac{\partial P''}{\partial x} &= 0 \\ \frac{\partial}{\partial t}\rho w + \frac{\partial P''}{\partial z} &= \rho B \\ \frac{\partial}{\partial t}\rho B + \rho w N^2 &= Q \\ \frac{\partial}{\partial x}\rho u + \frac{\partial}{\partial z}\rho w &= 0 \end{aligned} \tag{4.32}$$

where we assume as did Nicholls that the heat source has the form

$$Q = Q_o \rho \left(\frac{a^2}{a^2 + x^2} \right) \sin(mz) \tag{4.33}$$

An expression for the perturbation pressure can be derived.

$$P''_{xxtt} + P''_{zztt} + N^2 P''_{xx} = 0 \tag{4.34}$$

The deviation from Nicholls' hydrostatic solution is given by the nonhydrostatic term P''_{xxtt}. This causes the system of equations to be dispersive. Assuming a rigid lid at the top and bottom of the domain, it is possible to construct an analytic solution from the Fourier and Laplace transforms of this equation.

$$\hat{P}(k, z, t) = -\left(\frac{a e^{-a|k|} m \rho_o Q_o \cos(mz)}{kN (k^2 + m^2)^{1/2}} \right) \sin\left(\frac{Nkt}{(k^2 + m^2)^{1/2}} \right) \tag{4.35}$$

While it is not possible to analytically transform this equation back into real space via a small closed form solution, one can calculate this transform to a high degree of accuracy by approximating it with a discrete Fourier transform. This numerical form is subject only to the spectral errors of the spatial transformation. It is analytic in time and in the vertical direction.

A comparison of the spectral solution and the numerical model was done for the choice of parameters

$$a = 500\text{m}, \qquad m = \frac{\pi}{H}, \qquad H = 4\text{km}, \qquad Q_o = \left(\frac{g}{C_{pd} T_{\text{ref}}} \right) \left(2 \frac{w}{m^2} \right). \tag{4.36}$$

The domain was a 4x20 km domain with a resolution of 67 meters. The results of the comparison are shown in Figure 4.3. The lefthand column of this figure shows the numerical fields for the pressure, buoyancy and velocities and the righthand column is the spectral solution. Both columns agree well with each other.

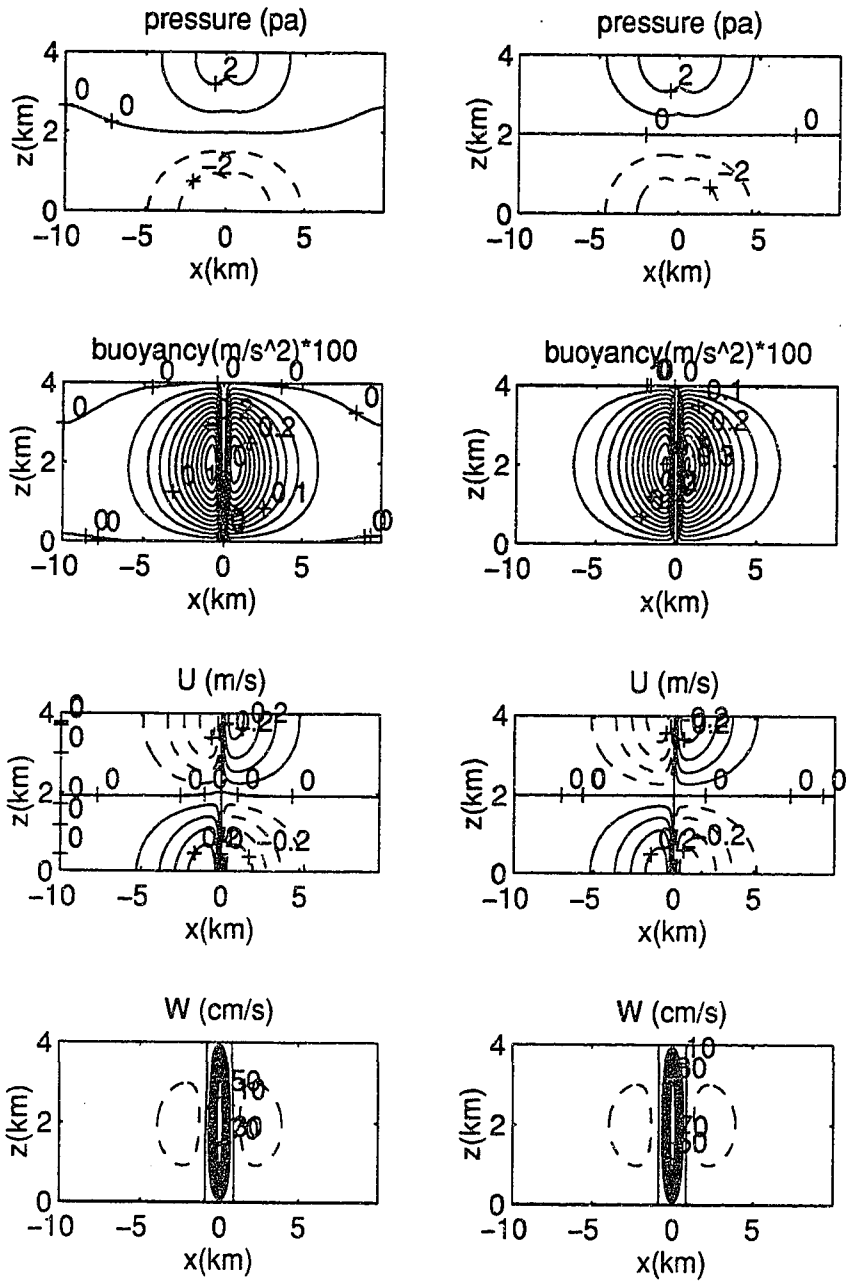

Figure 4.3: Comparison of the numerical (left) and linear spectral (right) solutions for a gravity wave test problem.

5 Multilevel Refinement

5.1 Introduction

A common problem of simulations of atmospheric convection is resolving turbulent motions. If a three-dimensional model is uniform in resolution, the number of gridpoints increases as $O(h^{-3})$, where h is an estimate of the grid spacing. This results in an increase in floating point operations of $O(h^{-4})$, as the timestep for an explicit model is limited by increases in spatial resolution. For three-dimensional models, this is a severe constraint. If one is to refine by a factor of n everywhere, the computational requirements increase as $O(n^4)$. One can then see that for a second order accurate method, the error of the solution will only decrease as $O(n^{-2})$. Hence, the amount of work increases quadratically for a given decrease in the error. Fortunately, by selectively refining only those regions of the domain that require more resolution, one has the promise of getting a $O(n^{-2})$ decrease in the error with an $O(n^2)$ increase in the amount of resources required. This is assuming that the regions of refinement have roughly the same number of gridpoints as the base domain.

Decomposing the domain into a sequence of nested grids called a composite domain is one way of achieving this goal. There are many different forms of composite grids in use. The simplest possible form is a hierarchical finite element method that decomposes a coarse triangulation of a domain, by subdividing elements into subelements. Efficient grid refinement for a finite difference scheme is a subtle problem which is actively being researched. The composite domain discussed in this thesis is a domain composed of nested grids that are aligned with the coordinate directions. These nested grids are composed of subdivided cells of the original coarse grid. There are two objectives when constructing a numerical algorithm for fluid flows in such a composite domain. One wants to create a comprehensive numerical treatment of the composite, while maintaining the advantages of a uniform discretization. We now discuss two numerical methods that have been proposed for flow solvers on a composite grid.

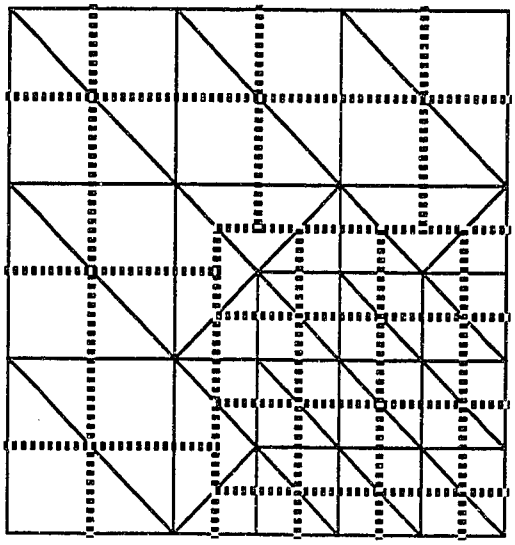

Figure 5.1: Illustration of FVE grid for a lower right hand corner. The solid lines indicate the FE lines and the dashed the finite volume.

The FVE method (McCormick 1989) provides a comprehensive treatment of block composite algorithms. The lower right hand corner of a domain is discretized using this technique in Figure 5.1. This was chosen to illustrate the interior and boundary discretization of the composite mesh. The volumes of the fine and coarse cells are constructed first in a regular way with the interface coarse cells dictated by the neighboring cells. The fluxes through the edges of the cells are calculated from a triangular finite element method. This insures that the discretization is conservative and hence preserves physical properties of the solution. This is important for solutions exhibiting sharp gradients or shocks where discretizations that rely on continuity of the solution break down. A regular triangulation is used for the coarse and fine gridpoints away from the interface. The triangulation of the interface gridpoints is found by opposing triangles to avoid coalescing grid lines with volume boundaries. By eliminating any coalescence of grid lines and cell boundaries, we insure that fluxes can be properly discretized. A useful feature of the composite triangulation is that it is a refinement of the original coarse grid. This has the effect of nesting the coarse grid triangulation in the finite element space of the composite. The

FVE composite suffers from an irregular treatment of internal cell boundaries and creates implicit discretizations. These features become cumbersome when applied to time-dependent problems.

Another composite method is the AMR (Automatic Mesh Refinement) composite. This is a cell-centered technique based on a finite volume interpretation of the cells. This method was originally formulated for hyperbolic equations (Berger and Oliger 1984) and later extended to incompressible flows (Almgren et al 1993). When a region of cells is refined, those cells are subdivided by an integer refinement factor. The composite mesh for the corner in Figure 5.1 is shown in Figure 5.2. There are no irregular cells on the border of refinement in this formulation. This permits the use of existing cell-centered uniform grid flow solvers. It also generalizes easily to an arbitrary number of levels of refinement and to complex systems of equations. Some of the analysis available from FVE can be used for an AMR composite when an odd refinement factors is used. The corner of Figure 5.1 with a refinement factor of three is shown in Figure 5.3. The regular cells of AMR are supplemented by the coherent grid structure of FVE. Grid lines and cell boundaries do not coalesce and the coarse grid is a coarsening of the composite. This enables finite element ideas to be used for constructing discretizations and simplifies the transfer of information between coarse and fine grids. It permits all grid points for a given grid and grid staggering to be referred to by a global index systems. This is especially useful for manipulating a staggered MAC grid.

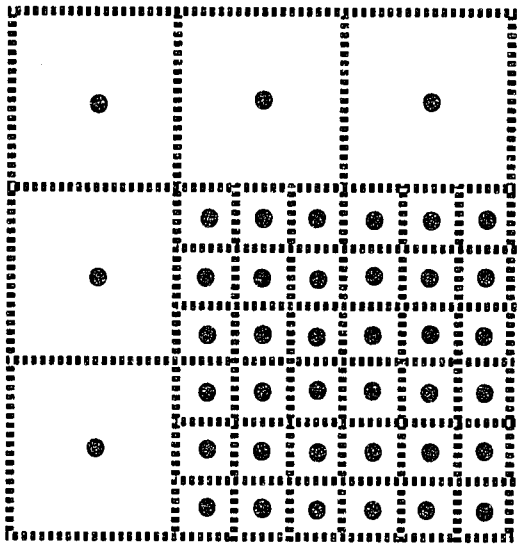

Figure 5.2: Illustration of the AMR composite for a lower right hand corner. The dashed lines indicate the borders of cells and the dots indicate the gridpoints located in the center of the cells.

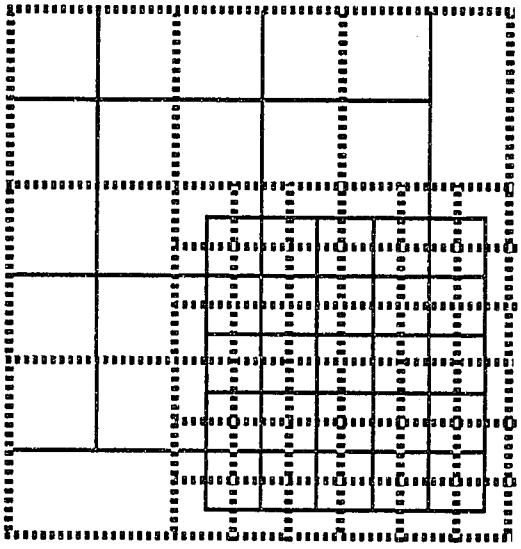

Figure 5.3: Illustration of the AMR composite for a lower right hand corner. The Solid Lines indicate grid lines and the dashed the borders of the cells. The intersections of grid lines represent gridpoints. The cells of the Berger and Collela grid have been retained, while the coherent grid structure of the FVE method is still present.

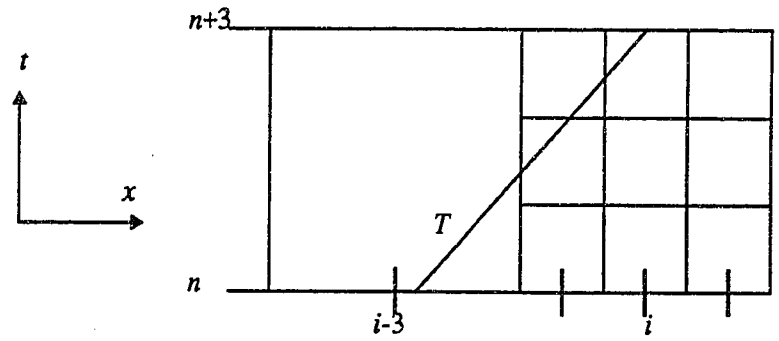

Figure 5.4: Illustration of a parcel trajectory T passing through an internal grid interface. By refining in space and time the parcel trajectory doesn't pass completely through a refined cell in a timestep.

The AMR composite uses refinement in time as well as in space (Figure 5.4). This is necessary for both stability and accuracy, since we are using an explicit algorithm for the advection. The Courant number of the flow remains constant when we refine temporally and spatially by the same ratio. This algorithm can be explained in the following steps:

1. Integrate a coarse timestep to provide boundary conditions on the inner domain for the intermediate timesteps.
2. Integrate 3 fine timesteps.
3. 'Restrict' or average fine domain information onto the underlying coarse cells.

This algorithm is easily extended by recursion to multiple levels of refinement. The multi-level aspects of this algorithm can be broken into two parts, the boundary conditions for the fine domain and the restriction of the fine domain to the coarse domain.

5.2 Multilevel Advection

One can use conservation of the global integral of ψ to determine a natural restriction for cell centered scalars. We insist that the volume integral over all the ψ in a grid Ω is constant except for boundary fluxes $\vec{F}_i$.

$$\int_{\Omega} \psi dV \approx \sum_{i \in \Omega} \psi_i dV_i = C - \sum_{i \in \partial\Omega} \vec{F}_i \cdot \vec{n}_i. \tag{5.1}$$

A simple form of restriction that satisfies this constraint on the interior of fine domains is the volume averaging

$$\psi_c = \frac{1}{n^3} \sum_{i \in \Omega_c} \psi_{fi}, \tag{5.2}$$

where Ω_c is the coarse grid cell containing ψ_c and n is the integer ratio of refinement. To maintain conservation in the presence of internal interfaces, we must enforce a relationship between the coarse and fine fluxes at the grid interface $\partial\Omega$

$$F^n = \frac{1}{n^3} \sum_{m=0}^{n-1} \sum_{i \in \partial\Omega} f_i^{n+m}. \tag{5.3}$$

Here capital letters indicate coarse grid variables and lower case letters indicate fine grid variables. The FVE method enforces this condition by design. The AMR method (Berger and Collela 1984; Berger 1987) involves refluxing the neighboring coarse cell with a corrective flux. This corrective flux is the difference between fine and coarse interface fluxes. An approach used by Clark and Farley (1984) is to replace the fine interface flux by a coarse flux that is interpolated spatially by a conservative interpolation formula. A final approach is to rely on continuity of the solution, or in other words to do nothing explicitly to enforce this condition. Through linear advection tests, we found that the AMR approach and the do nothing explicit approach were roughly equivalent in accuracy (Figure 5.5) and the do nothing explicit approach was approximately conservative (Figure 5.6). The refluxing method was a close second. For flows where the velocities are very turbulent, refluxing the neighboring cells might be required to maintain conservation. For flows almost tangential to the interface, the Clark and Farley method was found to be slightly unstable. This was due to the downwinding of advective fluxes that results from the interpolation formula.

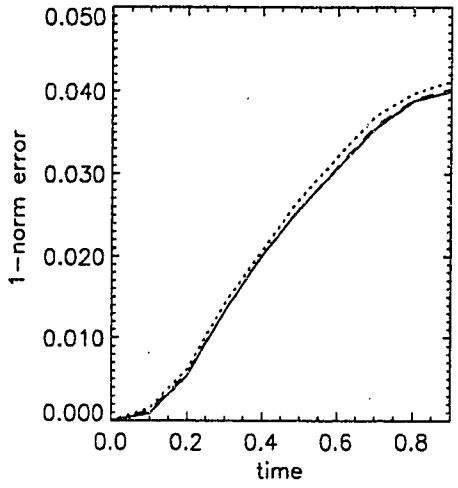

Figure 5.5: Error versus time of a 2D test of a cylinder in solid body rotation in and out of an interior refined grid. The cylinder completed one revolution exiting the refined grid at $t = 0.2$ and entering at $t = 0.8$. The solid curve is AMR, the dashed is do nothing and the dotted is Clark and Farley.

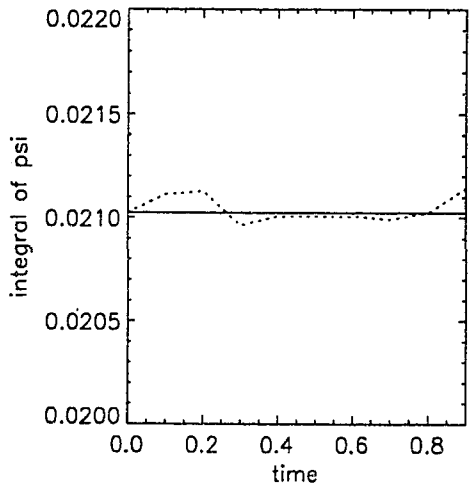

Figure 5.6: Conservation versus time corresponding to Figure 5.5. The solid curve is AMR, the dashed is do nothing and the dotted is Clark and Farley. The do nothing approach is approximately conservative.

5.3 Multilevel Continuity

Our formulation of the momentum equations was derived based on continuity of mass. Since conservation of mass is so fundamental, we require the restriction of velocity information to conserve mass. The MAC grid staggering provides a simple mass conserving formula. For the east face $\partial\Omega_e$ of a coarse cell, this formula is given by,

$$U_e = \frac{1}{9} \sum_{i \in \partial\Omega_e} u_i. \tag{5.4}$$

and shown in Figure 5.7. We restrict fine values of velocities to the coarse values by averaging over all the fine velocities that are coplanar with a coarse velocity component. This insures that if mass is conserved within the fine domain, the restriction will preserve this property in the underlying coarse domain.

Discrete mass conservation must hold at grid interfaces, i.e. the sum of the fine grid normal mass fluxes must equal the coarse grid mass fluxes at an interface. This prescribes a Neumann boundary condition on the inner grid pressure. The inner grid pressure perturbation ϕ satisfies the Poisson equation (4.9),

$$\frac{\Delta t}{2} \nabla \cdot \rho \nabla \phi = \nabla \cdot \rho \vec{U}^*, \qquad \vec{U} = \vec{U}^* - \frac{\Delta t}{2} \nabla \phi \ . \tag{5.5}$$

We use the same MAC grid stencil for the pressure in the refined domain as the coarse one, and we solve the resulting Poisson equation with an FFT-based direct solver similar to the one used for the global outer domain. This boundary treatment for the pressure was earlier formulated by Clark and Farley (1984). This was one of the first papers to illustrate the use of a mass conserving restriction and interpolation technique. The Neumann boundary conditions on the pressure allow the coarse and fine domain pressures to differ by an arbitrary constant. We remove this constant by subtracting from the fine domain the average difference of the fine domain pressure from the underlying coarse pressure. Figure 5.8 shows that with this correction, the pressure is continuous across all parts of the interface. The details of the simulation are given in section 5.4.

Enforcing continuity of mass in a time dependent model at fine grid timesteps requires not only spatial but also temporal interpolation of the normal mass fluxes imposed on it by the parent domain. The total flow rate through the side of a coarse grid cell that is parallel to the y-z plane during a coarse timestep can be expressed as the integral

$$F = \int_0^{\Delta t} \int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} \int_{-\frac{\Delta z}{2}}^{\frac{\Delta z}{2}} U\left(x_{i+\frac{1}{2}}, y_j + y, z_k + z, t\right) dz dy dt \ . \tag{5.6}$$

Our algorithm conserves a discrete analog of F through the internal boundaries. The temporal and spatial character of this integral is shown in Figure 5.7. The left-hand plot illustrates the temporal interpolation of mass and the right-hand plot illustrates the spatial interpolation.

If the spatial interpolation is conservative, linear interpolation in time will conserve the mass flux through this face:

$$\begin{gathered} u^{n+1} = \frac{2}{3}U^n + \frac{1}{3}U^{n+3}, \qquad f^n = \frac{\Delta t}{3}\left(\frac{u^n + u^{n+1}}{2}\right), \\ F = \frac{\Delta t}{3}\left(f^n + f^{n+1} + f^{n+2}\right), \\ F = \Delta t\left(\frac{U^n + U^{n+3}}{2}\right). \end{gathered} \tag{5.7}$$

The fine velocities on the normal face of a coarse cell at the interface must average to the value of the coarse velocity for conservative spatial interpolation,. This algorithm uses a form of quadratic interpolation that is based on the interpolation formula

$$\begin{aligned} u_f(\xi, \eta, \zeta) = {} & U + \xi U_\xi + \eta U_\eta + \xi\eta U_{\xi\eta} + \xi^2 U_{\xi\xi} + \eta^2 U_{\eta\eta} + \\ & \xi^2\eta U_{\xi\xi\eta} + \xi\eta^2 U_{\xi\eta\eta} + \xi^2\eta^2 U_{\xi\xi vv}, \end{aligned} \tag{5.8}$$

where the (ξ, η, ζ) coordinate system is the integer coordinate system of the enclosing coarse domain and the partial derivatives are evaluated with centered differences. The ξ^2,

η^2, and $\xi^2\eta^2$ terms all have nonzero sums over the coarse cell face, so the true interpolation formula used is

$$u(\xi,\eta,\zeta) = u_f(\xi,\eta,\zeta) - a_1 - a_2 - a_3 , \tag{5.9}$$

where the a_i represent the sum of these terms over the coarse cell. This formula was used as we tried to include as much quadratic information as possible. The combination of (5.7) and (5.9) insures mass continuity. Although this interpolation formula is slightly unstable for an advective flux due to downwinding, there is no stability problem due to the elliptic nature of mass conservation. For the case of variable density, the appropriate density weighting insures continuity.

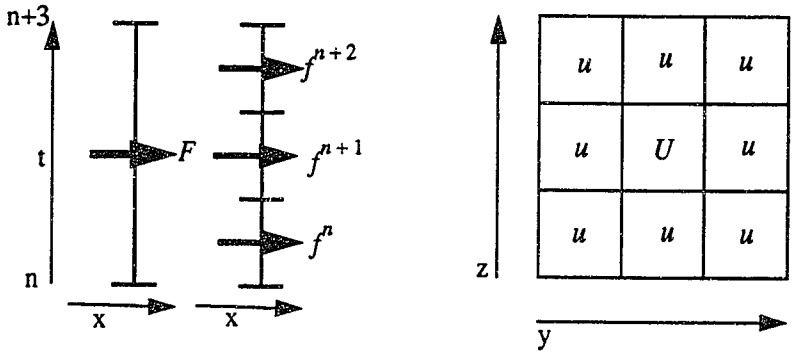

Figure 5.7: Sketch of conservation of mass at boundaries. Capital letters indicate coarse domain values. In the left plot, the coarse grid mass flux F is linearly interpolated in time to provide mass fluxes f at the midpoints of the three fine grid timesteps. The right plot shows the fine grid points u used in restricting for the coarse velocity U. The spatial interpolation formula preserves this property.

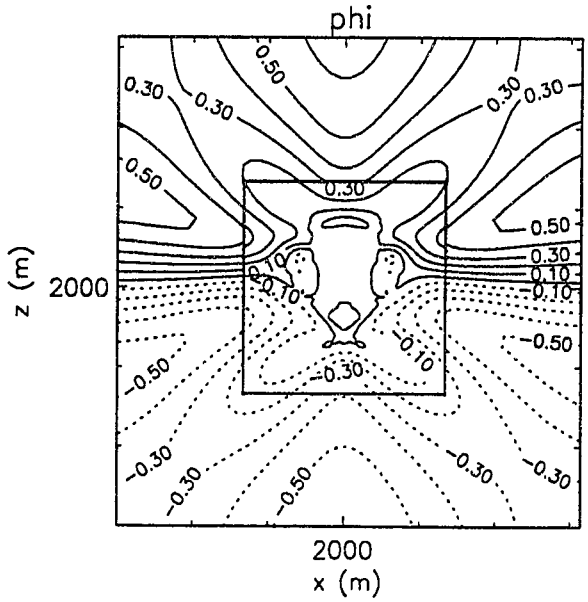

Figure 5.8: Normalization of fine grid pressure to match the coarse grid.The units of the pressure are Pascals and the contour interval is .1 Pascal.

5.4 Model Validation

The bubble collapse experiment of Section 4.4 was repeated with a single nested domain. As the buoyant bubble rises to its level of neutral buoyancy, it excites gravity waves that propagate throughout the domain. This test measures the ability of the internal boundary conditions on the pressure to transmit internal gravity waves back and forth.

We compared a single refined domain against two unrefined control simulations. These control simulations were 180 by 180 gridpoints and 60 by 60 gridpoints. The nested simulation consisted of a 60 by 60 base domain with a refined domain centered in the middle of the domain. The accuracy of the multilevel treatment can be seen by looking at the deviation in potential, kinetic and total energy from the control simulations to the multilevel simulation (Figure 5.9). There is no spurious creation or reflection of gravity wave energy by the treatment of the boundary conditions. The velocity interpolation formula

can be seen to create no oscillations at the boundary of the refined domain in contour plots of the vertical and kinetic energy (Figure 5.11). Overall, the nested algorithm captures about 75% of the accuracy improvement of the 180 by 180 simulation at approximately 14% of the computational cost.

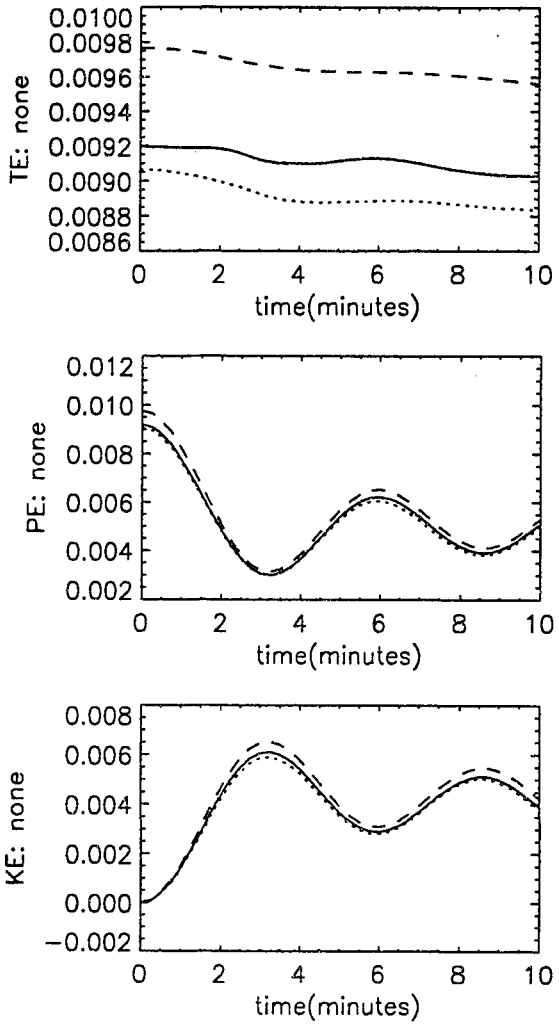

Figure 5.9: Comparison of energies between the control and refined simulation. Plots of fine control simulation (dashed), coarse control simulation (dotted) and the refined (solid) are shown.

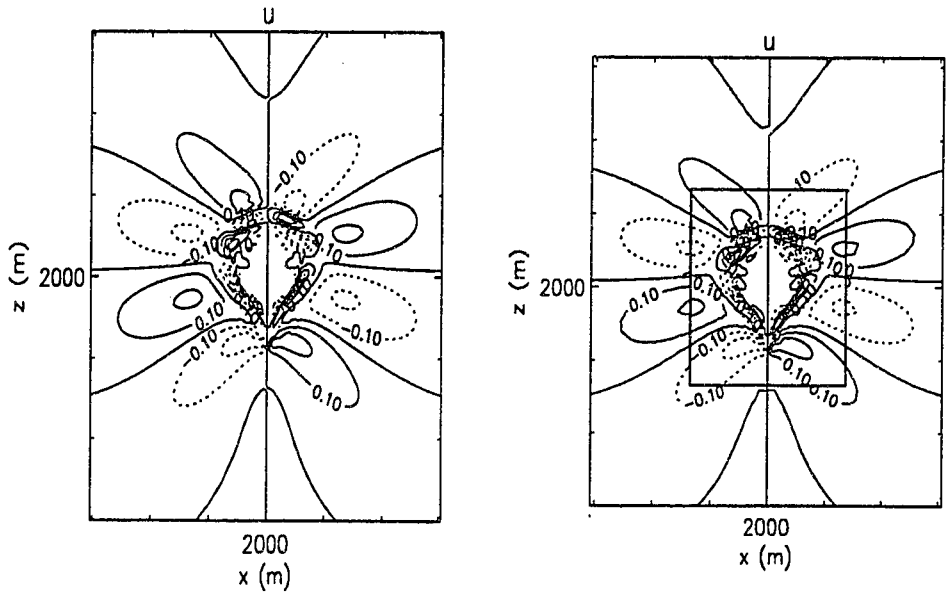

Figure 5.10: Horizontal velocity after 10 minutes for the fine control and refined simulations. Units are m/s and the contour interval is .1 m/s.

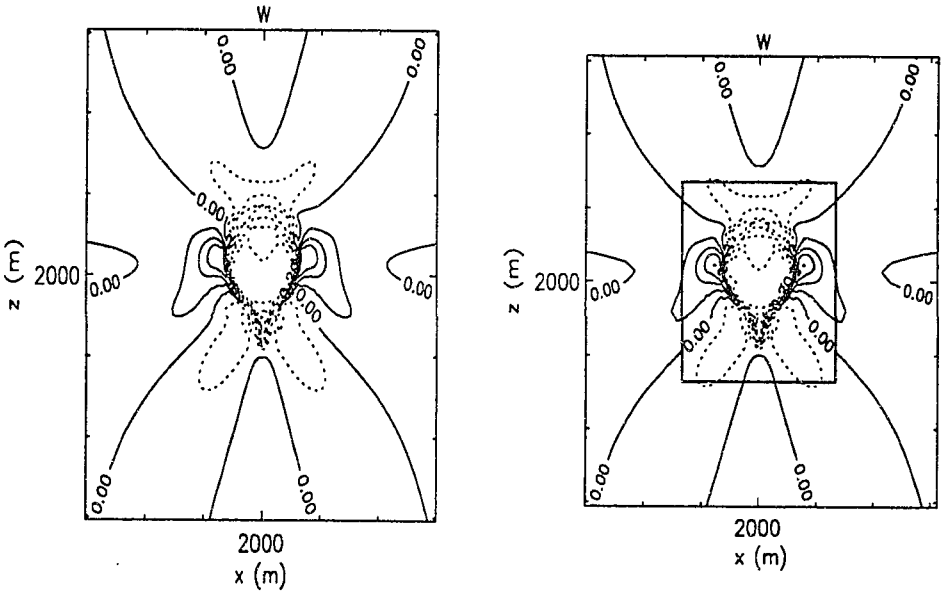

Figure 5.11: Vertical velocity after 10 minutes for the fine control and refined simulations. Units are m/s and the contour interval is .1 m/s.

6 Adaptive Refinement

6.1 Object-Oriented Numerics

A complex mathematical model can often be broken down into a hierarchy of levels of abstraction. This hierarchy enables one to choose the appropriate level of abstraction for the task at hand. Different ideas are included or left out depending on the purpose of the abstraction. Each level of abstraction focuses on different details. By considering an abstraction of a concept, we can reduce the complexity of the overall task to a manageable level. Numerics can be thought of as the implementation of mathematical abstractions on the computer.

Object-oriented numerics seeks to enable the modeler to utilize abstractions directly in the formulation and implementation of complex models. This is basis for the definition of the computer science concepts, class and object. An object consists of a current state and a set of operations that manipulate that state. A class is the definition of what the state of an object consists of and what the allowable operations are. Classes represent abstractions and an object is called an instance of an abstraction. The process of creating an object is called instantiation. Encapsulation is the concept of binding data and function together in an object. Encapsulation creates well-defined interfaces that represent and simplify using an abstraction. This interface and the ability to embed objects inside of objects allows the modeler to create a hierarchy of abstractions (Vermeulen 1992; Budd 1991; Lippman 1993).

Specialization of a model to a particular situation requires flexible interfaces of the abstractions used to create it. Object-oriented numerics provides a solution to the dilemma of creating general models, while being able to specialize, with the concepts of inheritance and polymorphism. Inheritance is the ability to create a specialized class from a general class. This ability enables one to create specializations, while at the same time retaining all of the abilities of the general class. Polymorphism is the ability of a specialized class to appear as a general class in applications that were written for the general class.

There are two general types of classes: Polymorphic classes that represent abstract concepts like algorithms and value-semantic classes that represent concrete concepts such as arrays (Wong et al 1993). Polymorphic classes usually form the higher levels of abstraction in a model. Inheritance is used to specialize the class to a particular application. The state of an object is changed by passing messages between objects. These messages may be interpreted differently by a general and specialized class of the same type. These messages are passed by calling member functions. Value-semantic classes make up the lower levels of abstraction and are associated with an algebra such as vector or matrix algebra. They are associated with a concise notation in which addition, multiplication, equals or other operations are defined. Abstract algebra is often a good tool to test the consistency of a value semantic class. These classes are used in the innermost loops of a mathematical model and their efficiency is critical. These properties are summarized in Table 6.1.

Table 6.1 Properties of Classes (Wong et al 1993, p. 661)

Property	**Polymorphic**	**Value-Semantic**
Hierarchy	Inheritances	Conversion
Interface	Message Passing	Algebra
Mechanism	Member functions	overloaded operators
Conceptualizing	Abstract	Concrete
Specialization	Numerous	Few
Efficiency	Not Critical	Critical

To use an object-oriented numerical approach, one should use the computer language that best implements polymorphic and value-semantic classes. Computer languages differ in the type of class they best support. Languages such as C++ and Smalltalk support polymorphic classes very well, but can often be inefficient for low level numeric computation. FORTRAN has a natural array syntax and is computationally efficient. Supporting efficient value-semantic classes is one of the driving motivators of FORTRAN 90. However, FORTRAN and its variants do not implement polymorphism and are cumbersome to use for complex data management. Complex numerical models require both types of classes,

polymorphic classes to provide flexibility and efficient value-semantic classes to perform the low level computations.

Rather than define a domain algebra for my multilevel method as in Hilfinger (1989), I implemented a hierarchy of polymorphic classes in C++, which overlay a layer of FORTRAN subroutines that perform the base numerics. The resulting hybrid was able to use the strong points, while being free from the deficiencies of both. The base of my abstraction hierarchy is the concept of a Brick that represents a region in three-dimensional space. An abbreviated definition of this class is shown in Figure 6.1

```
class Brick {
    int scale;                  // depth of refinement
    int NX,NY,NZ;               // dimension extents
    int iorigin,jorigin,korigin; // Position of (imin,jmin,kmin) in the global system
    int ishift,jshift,kshift;   // the grid staggering
    int imin,jmin,kmin;         // the minimum index
    int imax,jmax,kmax;         // the maximum index


    void CreateImages();
    void Divide( Brick&, Brick&, int, int );          // Divide in two
    void Restrict( Brick, Brick, float *, float * );  // restrict fine to coarse
    void Fill( Brick, Brick, float *, float *);       // interpolate coarse or fine to fine
    void Display();                                   // display the Brick
    int Contained( Brick );                           // test for containment
    Brick Intersection( Brick, int );                 // compute the intersection
};
```

Figure 6.1: Abbreviated description of type Brick. int is short for integer and float is short for a floating point number. The state of the object is given by its members and the allowable operations on the state. The member functions are listed below the data members.

This class is built about the formula

$$i_g = R\,(i - i_{\min}) + i_{\text{origin}} \tag{6.1}$$

that holds for any odd ratio of refinement regardless of grid staggering. The index i_g is a position in a global index space at the highest level of refinement. i_{origin} is the global

index position of the local grid index i_{min}. An important advantage of refinement by odd numbers is that gridpoints at all levels of refinement are always coincident with gridpoints at the highest level of refinement. This makes it possible to perform interpolation and restriction with fast efficient array operations between arbitrary Bricks of arbitrary grid staggering. The other member functions extend the interface by performing general display and diagnostic tasks.

Our adaptive multilevel method is simplified by the process of message passing. The task of implementing adaptivity in our method consists of taking an array of domains representing a uniform block composite, computing an error diagnostic and using this diagnostic to create a more optimal array of domains. A simple hierarchy is given by a class called Regrid that takes an object of type Composite that represents the geometry of the initial composite domain and returns a new Composite object that represents the optimized composite. Inside of Regrid, each level of the multilevel algorithm is created by creating a Cluster object. Cluster allows Regrid to be formulated recursively by freeing it from the actual details of the clustering algorithm. Inside of Cluster, objects of type BBox are used to formulate the geometry of each of the refined grids. Each step in the process from high to low level has been encapsulated to preserve its organization.

Polymorphism allows us to create a general integrator for integrating our system of equations. Otherwise, this breakdown of tasks, although necessary, would be very inflexible. We create AfieldComp and PfieldComp as general classes for advecting a quantity or computing a pressure. Each of these classes manipulates an array of Afield or Pfield objects that are responsible for solving advection or elliptic equations on a domain. Inheritance is used to implement specific numerical operations. The inheritance hierarchy for advection on a domain is shown in Figure 6.2. The purpose of the hierarchy is to advect a quantity from time level n to time level n+1 in a general way. The base class Brick provides the spatial extent of the gridpoints in the domain. The derived class Afield contains a Volume which is a Brick with data that represents the solution at time level n. This volume might contain fictitious image points that aren't related to the computational extent of the

domain. It might contain Sarray objects which are Volumes that represent slave nodes for internal boundary conditions. Afield has not been specialized to a particular algorithm for advecting a quantity. This class is only useful in that it provides a general interface by which a derived class can be used. For instance, we can define class SL as the derived class of Afield that uses the forward-in-time advection scheme presented in Chapter 2. SL is a specialized class that is able to use all of the basic abilities that have been defined for Afield without having to modify any of the code that has been created for Afield. We want to implement specialized classes for other advection methods to compare with our method. Polymorphism allows us to write code for the base class Afield and then use it for a specialized derived class. This simplifies the implementation of the class AfieldComp. This class is a container class that manipulates a general array of Afield objects over a composite domain without knowing which particular advection scheme is being used.

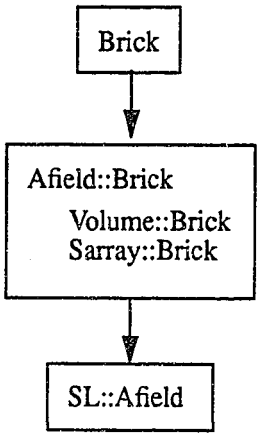

Figure 6.2: Advection algorithm inheritance tree. Inheritance is shown by (derived class::base class).

6.2 Adaptive Refinement

There are strong demands on the clustering algorithm of an adaptive method. It is conceivable that the composite may be regridded every few timesteps for a rapidly varying flow. This requires an algorithm that is computationally cheap and creates composites that

retain the advantages of a block composite algorithm. In addition, grids may need to have dimensions that are the products of small prime numbers to use modern elliptic solvers. We use a clustering algorithm that was originally formulated by Berger and Rigoutsos (1991). The user must specify a criterion to determine if a cell should be marked for refinement. Criteria that we have used for simulating boundary layer cloud simulations are the magnitude of the vorticity and presence of liquid water. Other researchers have marked cells using Richardson extrapolation to determine the truncation error for a given field in a cell (Berger and Collela 1984). We denote marked cells by setting an array $M_{i,j,k}$ to unity in marked cells and zero everywhere else. Berger and Rigoutsos used a technique from image processing in which marked cells are clustered into refined domains using natural breaks in the histograms of $M_{i,j,k}$ that are compiled in coordinate directions. The hole histogram

$$H_i = \sum_{\substack{1 \le j \le \mathrm{NY} \\ 1 \le k \le NZ}} M_{i,j,k} \qquad 1 \le i \le \mathrm{NX}, \tag{6.2}$$

is used to calculate gaps between regions of marked points in the x direction for a domain of size (NX,NY,NZ). Unneeded refinement on the edge of this domain is represented by a segment of zeros on the end of this histogram. Segments of zeros on the interior of this histogram represent potential places to split this domain. Transitions between regions of marked points and regions of unmarked points in the x direction are found by an edge detection routine using the x edge histogram E_i.

$$E_i = \sum_{\substack{1 \le j \le NY \\ 1 \le k \le \mathrm{NZ}}} (M_{i,j,k} \ \mathrm{XOR}\ M_{i+1,j,k}), \qquad a \ \mathrm{XOR}\ b = \begin{cases} 0 & a = b \\ 1 & a \ne b \end{cases}. \tag{6.3}$$

This histogram is the sum of the number of transitions between unmarked and marked points in the x direction. The location of suitable edges to split a domain is found from the x edge difference histogram D_i

$$D_i = E_i - E_{i-1} \tag{6.4}$$

Locations where this histogram crosses zero indicate lines of transition between regions of marked and unmarked points. which are good locations to put a domain boundary. The growth or shrinkage of E_i is a good indicator if a marked region is growing or shrinking in the x direction. The derivative of E_i changing sign indicates that this growth or shrinkage has stopped. The evaluation of these edge and hole histograms is computationally cheap and results in non-overlapping grids.

A requirement of many numerical techniques for solving elliptic equations is that the dimensions of a domain be products of small prime numbers. We insure that the dimensions of a refined grid are products of small prime numbers by imposing a uniform stride of length S in the edge and hole detection. This results in fine grids with dimensions given by

$$NX = nmS \tag{6.5}$$

where n is the refinement ratio and m is the number of strides in a given dimension. m can be kept to a reasonable size by cutting a domain in half, when it becomes too large.

We require that the final refined domains be computationally efficient. By this we mean that we try to refine as few as possible unmarked points. This insures that few computational resources are spent on regions where refinement is not needed. This is an important requirement for three dimensional calculations where a refinement factor of n creates n^3 fine cells for each coarse cell subdivided. The efficiency of a grid is defined by

$$\varepsilon = \frac{1}{(\mathrm{NX\ NY\ NZ})} \sum_{\substack{1 \le i \le \mathrm{NX} \\ 1 \le j \le \mathrm{NY} \\ 1 \le k \le \mathrm{NZ}}} M_{i,j,k} \tag{6.6}$$

if ε falls below a fixed tolerance $\varepsilon_{\mathrm{tol}}$, the grid is considered inefficient and is further subdivided. The optimal choice of $\varepsilon_{\mathrm{tol}}$ is a compromise between the creation of large numbers

of refined domains and unnecessary refinement. Empirically, it was found that $\varepsilon_{tol} = 0.8$ is an optimal value if a minimum number of efficient domains is desired. With $\varepsilon_{tol} = 0.95$, we minimize the amount of unnecessary refinement at the expense of creating many refined domains. Excessively optimizing the refined domains was found to increase computational work by increasing the frequency by which the domain needed to be regridded.

A general region of marked points can be clustered into refined domains by applying this algorithm recursively. This is illustrated by the algorithm's results for the U shaped domain in Figure 6.3. The initial candidate for refinement is found by eliminating the surrounding unmarked points around the central region as edge holes. Since this candidate is deemed inefficient, we try to find an internal edge or hole to subdivide it. If one can't be found we are done, otherwise we split it into two new candidates and discard it. This process is then applied for the new candidates. This process stops when no more splitting can be done or all of the candidates are deemed efficient.

This algorithm was tested on the three-dimensional advection of a sphere of uniform concentration with a U shaped region cut out of it in on a 50^3 domain. This was the adaptively refined version of the advection test in Section 3.4. The clustering was done with $\varepsilon_{tol} = 0.5$ every 10 timesteps. This efficiency was chosen to minimize the number of internal domains created. The criteria for refining gridpoints was whether the advected quantity ψ was above a certain threshold. A small buffer zone of three gridpoints was added to the marked region to insure that the sphere would not be immediately advected outside of the refinement region. A view of the composite halfway through the simulation is given by Figure 6.4. To estimate the effect of the adaptive refinement we compared its error against the error of two uniform resolution simulations of 50^3 and 80^3 gridpoints. The 80^3 simulation was near the largest uniform resolution simulation that computer memory limitations allowed. The growth of the error of these simulations is shown in Figure 6.5. We see that adaptive domain was able to halve the error of the 80^3 simulation.

The adaptive simulation took twice as long to compute and 20 percent more memory than the 80^3 simulation. The adaptive simulation was able to halve the error for less than twice the computational cost of the 80^3 domain. This illustrates that the adaptive multilevel method was able to increase the accuracy with only a linear increase in cost versus the quadratic or higher cost of increasing the resolution of a uniform mesh. This simulation was able to stretch the accuracy of our advection algorithm beyond the our computational limits for a uniform grid.

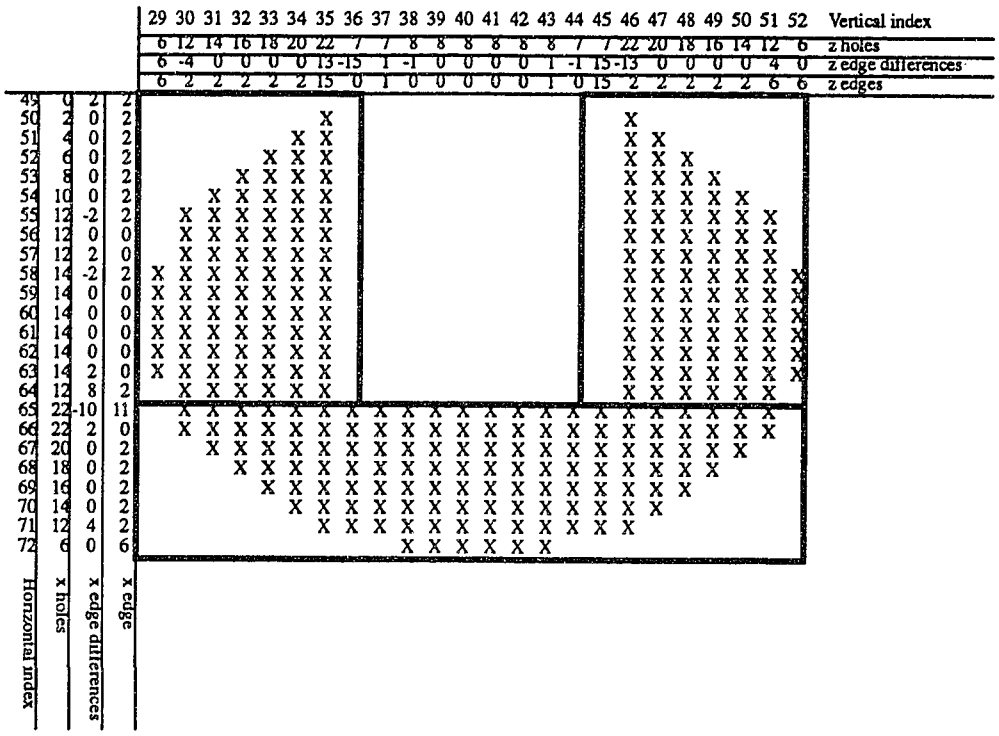

Figure 6.3: Clustering Algorithm. X's indicate marked points and bold lines indicate final grid boundaries. A stride of four was chosen.

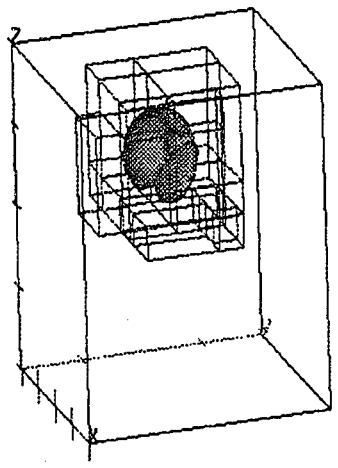

Figure 6.4: Illustration of the clustering algorithm for a three-dimensional advection test.

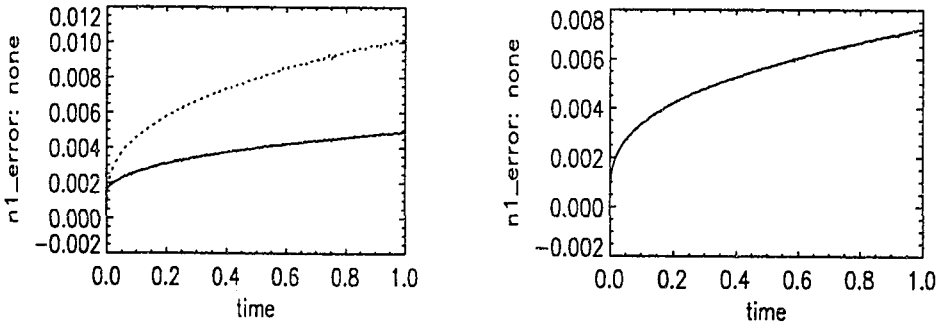

Figure 6.5: Comparison of growth of the 1-norm error for a 50^3 unrefined domain and a 50^3 refined domain. The right hand plot shows the error growth for a unrefined 80^3 unrefined domain.

7 Cumulus Convection

7.1 Dynamics

Numerical models are useful tools for understanding properties of the trade cumulus boundary layer. Comparing simulations with observed clouds lets us examine the interplay of physical processes that produced the convection. Simulations can provide statistics that are not readily available from observations and allow us to assess the sensitivity of a system to changes in parameters. In this chapter, we simulate cumulus clouds in varying environmental soundings to assess how the transport of mass, heat, moisture and momentum depend on the sounding.

The conditional instability and stratification of the environment affect the transport of low level air. The buoyancy of a cloudy parcel is determined by its virtual potential temperature deviation from the environment which is affected by mixing with environmental air. Latent heating affects this temperature deviation by releasing large amounts of heat to the parcel. Ultimately, all air parcels processed by the cloud move to their level of neutral buoyancy. The transport of heat and moisture by the cloud will in turn effect the stratification of the environment. A goal of simulating these clouds is to better understand and parameterize these effects for the turbulent ensemble of parcels within a typical cloud.

An adaptive multilevel method is well suited to simulating the fine scales associated with the mixing in a cumulus cloud and capturing the interaction of the cloud and the large expanse of stably-stratified environment in which the cloud is embedded. In the near vicinity of the cloud, it is important to resolve the buoyantly driven turbulent eddies. At the top of the cloud, it is particularly important to resolve the sharp moisture and temperature gradients that develop. The process of convection also tends to create strong gradients by the transport of warm cloudy air past cool dry environmental air. The cloud entrains and mixes environmental air across strong gradients at the trade inversion. Given adequate resolution, the forward-in-time algorithm presented in Chapter 4 is capable of accurately

simulating these features. The multilevel formulation of Chapter 5 enables this algorithm to attain this resolution.

The most important problem for numerical simulation of mixing processes is the representation of space and time scales that are unresolved. This process of modelling the flow of energy from resolved to unresolved scales is called turbulence parameterization. Mixing in a cumulus cloud is caused by buoyantly driven turbulence. Energy generated by buoyancy forces in larger eddies 'cascades' to smaller scales, first an inertial subrange of isotropic turbulence in which eddies are affected much more by inertial than buoyancy forces, then to very small eddies being dissipated by viscosity. A common strategy of turbulence parameterization is to identify where there is unresolved turbulence and design a scheme by which energy is passed down from resolved eddies into unresolved scales in a way that mimics the unresolved turbulence. Unfortunately, this strategy isn't always feasible. The flow can include large density gradients even at the smallest resolved scales. Turbulence parameterizations usually assume that the unresolved turbulence is in the inertial range and is nearly isotropic, an assumption which fails where the parameterization is most needed. One is still limited with an adaptive multilevel method by what one can resolve. However, an adaptive multilevel method can increase the range of resolvable length scales. The higher levels of refinement in a multilevel method reduce the importance of turbulence parameterization by lowering the cutoff of resolvable scales. The following simulations were done without the use of a turbulence parameterization. We allowed the numerical dissipation due to flux correction to handle this by itself.

The effect of turbulent mixing in a simulated two-dimensional cumulus cloud for the environmental sounding of Figure 2.1 is shown in Figure 7.1 and Figure 7.2. The cloud is initialized as a positively buoyant cylindrical bubble with the same thermodynamic properties as the air parcel discussed in Section 2.1. This bubble has an initial temperature excess of 0.5 K and a moisture excess of 0.3 g/kg. The sounding has a wind profile of 2 m/s per kilometer shear from the ground to the trade inversion. A cloud forms when the bubble reaches its level of condensation and rises until it hit the inversion layer. The evapora-

tive cooling on the edges of the cloud partially drives the production of eddies that rapidly proceed to mix the core of the cloud at later times.

This simulation illustrates the usefulness of an adaptive multilevel method. The simulation used a 12 km wide by 4 km deep outer grid to resolve the global flow in which the cloud is embedded and interactive inner grids of three times the resolution that were regridded every 6 timestep (30 seconds) to track the cloud. These inner grids are marked by dashed outlines in Figure 7.1 and Figure 7.2. The criteria for marking whether a gridpoint should be refined was whether there was cloud water present. A region of five coarse gridpoints around the cloud was also marked. The ability to partially refine the domain made it feasible to perform most of the work on the inner domain. The multilevel method made it possible to efficiently resolve both the flow in the area around the cloud and the fine-scale mixing. By using a composite of rectangular grids, a non-grid aligned feature was efficiently modelled.

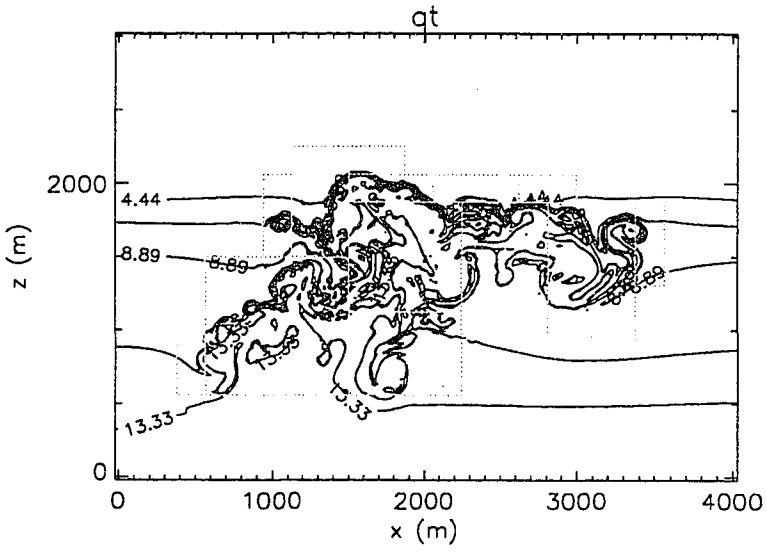

Figure 7.1: Total water mixing ratio for simulated bubble at t = 20 minutes.

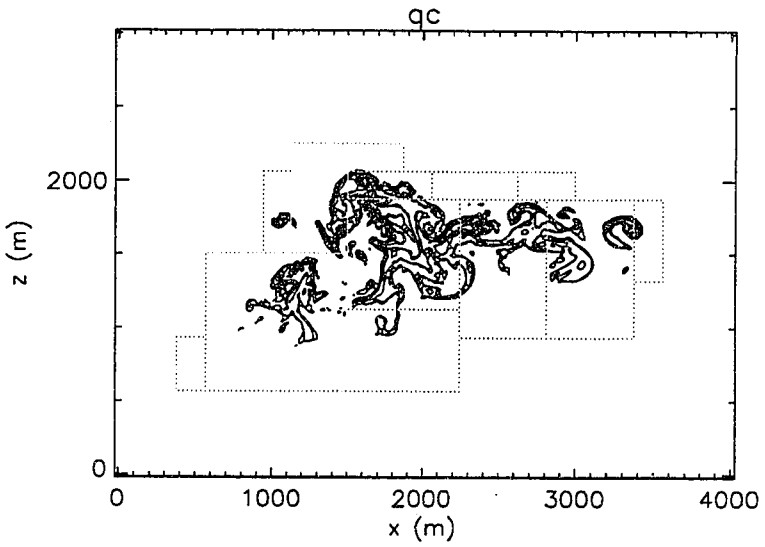

Figure 7.2: Liquid water mixing ratio for simulated bubble at t = 20 minutes.

In this chapter, we compare a numerical simulation with radar and aircraft observations of a Hawaiian cumulus cloud that was observed by the Hawaiian Rainband Project (HARP) during the summer of 1990. Some of this work is also discussed in a paper by Grinnell et al. (1995). During HARP, the height of the trade inversion ranged from 1.4 km to 3.7 km with an average mean height of 2.1 km. Cloud base tended to form at an altitude of around 500 meters (Grinnell et al. 1995). This resulted in a range of states from suppressed to active convection. Doppler radar observations were used to deduce vertical mass flux profiles for cumulus clouds for varying stages of evolution and cloud size. Grinnell et al. classified the observed clouds into large clouds, rainbands, and small isolated cells. We compute numerical mass flux profiles for a 'bubble'-initialized cumulus cloud and compare them to the observed profiles. In addition, we can make profiles of the cumulus induced vertical heat and moisture fluxes from the numerical model that are not available to the observations.

A cumulus capped boundary layer includes many convective clouds at various stages of their lifetimes collectively reacting to large-scale destabilization. To understand the collective effects of the clouds, we consider the vertical transports due to a typical cumulus cloud integrated over its lifetime. Many cumulus parameterizations also represent convection using a simple model cloud, whose transports should mimic the lifetime-integrated transports in a typical cumulus cloud. In addition to comparisons with an observed cumulus cloud, this chapter investigates the effect on transports of varying shear and relative humidity on the evolution and transport properties of a typical cloud in an idealized trade wind boundary layer sounding. Varying the relative humidity alters the evaporative cooling due to entrainment. Varying the shear alters the cloud dynamics and in particular the lateral entrainment processes.

7.2 Analysis of Entrainment, Detrainment and Mixing

We are interested in the bulk transport properties of a cumulus cloud. To analyze these properties, we define a control volume in which turbulent transports are occurring or have occurred. Since air outside the cloud is stably stratified, the cloud affected region (CAR) consists of air that is saturated or bears evidence of turbulent mixing. Undilute saturated air is the main source of latent heat to power the cloud. Unsaturated mixed air is assumed to have been processed by the cloud

Mixing is evaluated by a conserved variable analysis introduced to cloud physics by Paluch (1979). Our mixing analysis used two conserved passive tracers in the numerical model, both functions of initial vertical position,

$$\zeta = z/H \qquad h = \zeta^2, \tag{7.1}$$

where H is the height of the domain. This analysis is an extension of previous work by Smolarkiewicz and Bretherton (1989) where horizontal and vertical tracers of initial position were used to diagnose mixing and entrainment. The tracers (h, ζ) can be used to construct a "conserved variable diagram" where all gridpoints lie initially on the environ-

mental curve (h_e, ζ_e) as shown in Figure 7.3. These quantities are advected along with the flow and the only way a gridpoint can change its position on the conserved variable diagram is by mixing. Since the numerical method is conservative, mixing is a linear process and mixtures fall to the left of the environmental curve. These tracers were chosen since they are nonlinearly related. q_t and θ_l could also be used, but they are too linearly correlated to be useful in a trade cumulus sounding. Using these tracers, we computed a "mixing field".

$$D_{\text{mix}}(\hat{x}, t) = \min_{0 \le z' \le H} ((\zeta(\hat{x}, t) - \zeta_e(z'))^2 + (h(\hat{x}, t) - h_e(z'))^2) \tag{7.2}$$

The quantity D_{mix} is the minimum distance of a point (h, ζ) from the environmental curve (h_e, ζ_e) on the conserved variable diagram illustrated in Figure 7.3. Air is said to be irreversibly mixed if this field achieves a threshold of 1% of a typical maximum value. The mixing field (Figure 7.4) for the simulation in Figure 7.1 shows that the CAR extends further than the region that is diagnosed as saturated. This indicates the effect of cloud motions extends into the surrounding unsaturated air.

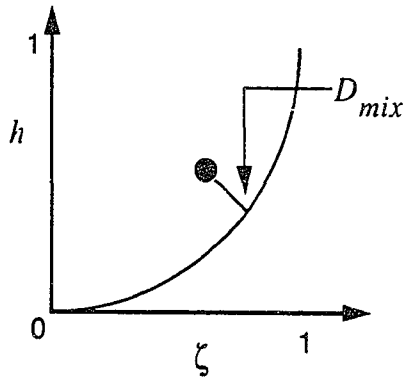

Figure 7.3: Illustration of a conserved variable diagram where the abscissa is the initial vertical height tracer and the ordinate is a function of the initial vertical height. The environmental curve represents the location of all unmixed parcels.

The environmental motions surrounding the CAR are predominantly horizontal in nature. ζ was found to deviate by only a few percent from z/H in the undisturbed environment. This aspect of cumulus convection is shown for a three-dimensional simulation of convection for a HARP sounding. Figure 7.5 shows an opaque outline of the CAR with an isosurface of constant ζ intersecting it. This isosurface is representative of all the isosurfaces that intersect the CAR. The CAR contains all of the turbulent motions that form the active convection, while the exterior of the CAR is stably stratified and includes only a laminar reaction of the environment to the cloud. The CAR provides a natural boundary for computing the transport properties of the cloud that rely on turbulent motions.

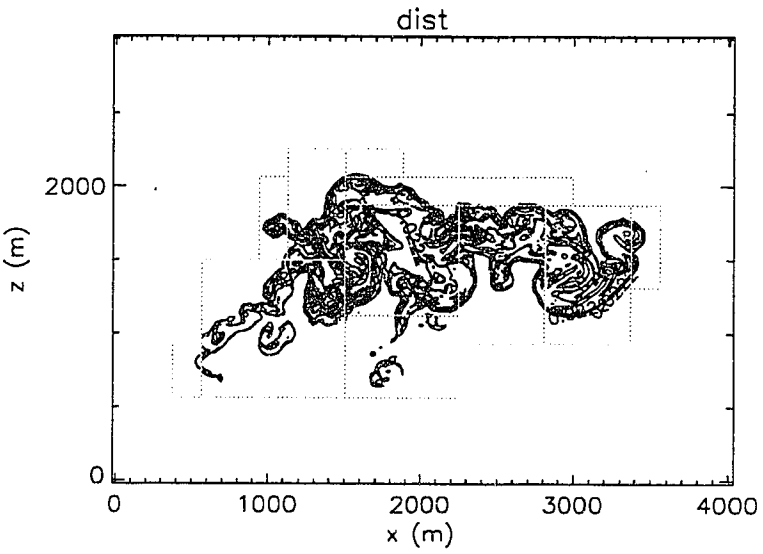

Figure 7.4: Mixing field overlaid by the liquid water field.

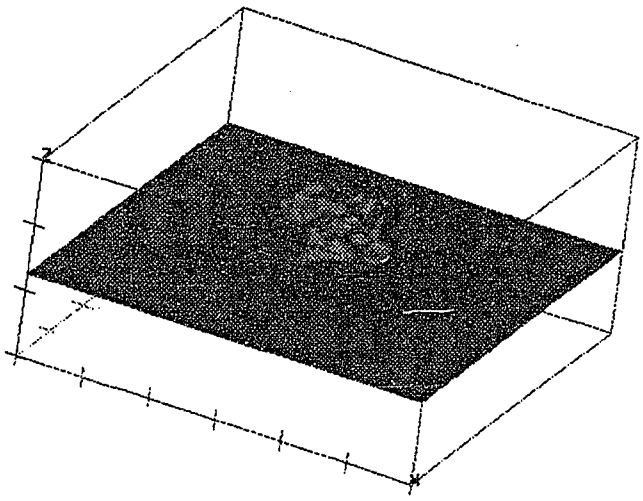

Figure 7.5: Contour plot of an isosurface of ζ intersecting an opaque outline of the CAR for a simulation of an Hawaiian trade cumulus. Each tick mark in the horizontal and vertical is one km.

We first examine the vertical transport of mass caused by the cumulus cloud. We can deduce the time-integrated vertical mass flux $M(z)$ and its associated time-integrated entrainment $E(z)$ and detrainment profiles $D(z)$. Entrainment is defined as the mass that is entrained into the CAR by turbulent motions and detrainment is defined as the final mass of air that has accumulated inside the CAR. This definition of "detrained" air is unconventional, since it includes air that is still part of the cloud. These profiles are related to each other by the mass balance

$$\frac{\partial}{\partial z}M = E - D \,. \tag{7.3}$$

Hence, from two of these quantities, we can deduce the third.

A useful profile is found from the vertical distribution of mass in the CAR at a given time,

$$D = \int_{\Omega_{car}} \rho dA \,. \tag{7.4}$$

Since mass is irreversibly mixed, mass can only enter the CAR and this profile can be thought of as representing the air that has been processed by the cloud at a given time. When the thermal has died, the final mass of the CAR can be thought of as detrained mass in the conventional sense. Figure 7.6 shows the distribution of D for the HARP cloud in Figure 7.5. We conclude that mass that enters the CAR at cloud base preferentially seeks a level just below the trade inversion. In addition, there is a significant amount of mass that detrains around 1 km, well below the inversion. This mass consists of evaporatively cooled mixtures of cloud and environmental air.

A first estimate of the time integrated mass flux caused by the cloud is found from the integrated mass flux over the CAR

$$M_w = \int_0^t \int_{\Omega_{car}(t')} \rho w dA dt' , \tag{7.5}$$

where $\Omega_{car}(t')$ is the area of the CAR at time t'. A mass balance based on M_w and D can be thought of as providing information on the vertical rearrangement of mass that occurs within the CAR due to the average vertical velocity of the turbulent motions. This is seen by the derived entrainment profile

$$E_w = \frac{\partial}{\partial z} M_w + D \tag{7.6}$$

In Figure 7.6, a large entrainment peak is seen at cloud base and is the result of all the cloud base air being entrained into the CAR at cloud base, even though it originated from lower levels. This entrainment profile represents the mass that is entrained where it enters the CAR and is unable to determine the original height of entrained air. The air that enters the cloud at cloud base may have originated well below cloud base and air that is entrained at a given height might have been first driven there by compensating motions caused by the cloud. To determine an estimate of the original height from which entrained air is drawn, we use passive tracers to derive an entrainment profile that is able to determine these origins.

Rather than trying to diagnose the air entering the CAR, we look at the difference between the initial mass of environmental air within a given range of ζ and the mass outside of the CAR within the same range of ζ at a later time. The difference between these masses is air that has been entrained into the CAR and has originated at heights within this range of ζ. This entrainment profile is found by initializing it to the total mass at a given level and subtracting off the mass that has not been entrained,

$$E_\zeta = L_x L_y \rho(z) - \int_{C\Omega_{car}} \rho(z')\delta(\zeta(x', y', z') - z)dx'dy'dz' . \tag{7.7}$$

$E_\zeta \Delta z$ represents the mass entrained from between original heights of $z - \Delta z/2$ and $z + \Delta z/2$. Numerically, this is accomplished by a sorting procedure. We initialize $E_\zeta(k) = L_x L_y \rho_k$ and the vertical tracer ζ is initialized as $\zeta_{i,j,k} = z_k$ at all heights z_k. At some later time, this tracer field has the value $\zeta_{i,j,k} = z_{k1} + \alpha$. where z_{k1} is the highest grid level below $\zeta_{i,j,k}$ and α is the remainder. If the grid point is diagnosed as lying outside of the CAR, its contribution to the entrainment profile becomes

$$\begin{aligned} E_\zeta(k1) &= E_\zeta(k1) - \Delta x \Delta y \rho_k (1 - \alpha/\Delta z) \\ E_\zeta(k1+1) &= E_\zeta(k1+1) - \Delta x \Delta y \rho_k (\alpha/\Delta z) \end{aligned} \tag{7.8}$$

By iterating over all of the gridpoints that lie outside of the CAR, we can sort the air that enters the CAR from each level. This sorting procedure can be expressed mathematically as

$$E_\zeta(k) = L_x L_y \rho_k - \sum_{i1}\sum_{j1}\sum_{k1} \Delta x \Delta y \rho_{k1} F(\zeta_{i1,j1,k1} - z_k) \tag{7.9}$$

where the delta function has been approximated by

$$F(\zeta) = \begin{cases} 0 & \Delta z < \zeta \\ 1 - \zeta/\Delta z & 0 < \zeta < \Delta z \\ 1 + \zeta/\Delta z & -\Delta z < \zeta < 0 \\ 0 & \zeta < -\Delta z \end{cases} \quad . \tag{7.10}$$

Associated with this entrainment profile is the vertical mass flux profile M_ζ obtained from the mass balance

$$M_\zeta = \int_0^z (E_\zeta - D)\, dz \ . \tag{7.11}$$

Because we have integrated over all gridpoints in the complement to the CAR, we know that E_ζ is always equal in net mass to the profile D. This insures that M_ζ will automatically asymptote to zero at cloud top consistently.

We compare the two mass balances in Figure 7.6 and Figure 7.7. These figures were generated by the example Hawaiian cumulus cloud of Figure 7.4. In Figure 7.6, the entrainment and detrainment profiles are compared. The entrainment profile E_ζ can determine the original height distribution of the mass flux into the base of the CAR, while E_w has a large entrainment peak at the base of the CAR. Both mass balances produce entrainment profiles that are nearly uniform above cloud base. This provides motivation for assuming uniform entrainment above cloud base as is done in the Raymond and Blyth (1986) model.

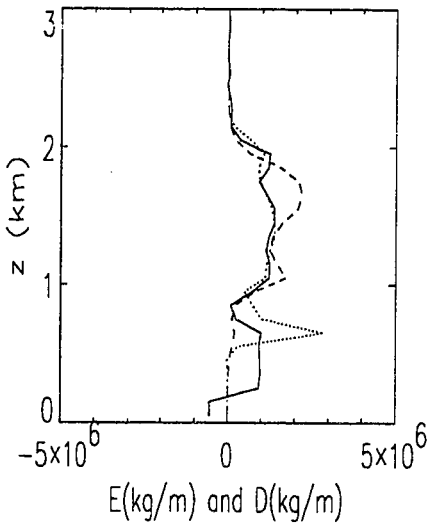

Figure 7.6: Comparison of the entrainment and detrainment profiles at t = 20 minutes for the Hawaiian simulation. The solid line is E_ζ, the dotted line is E_w, and the dashed line is D.

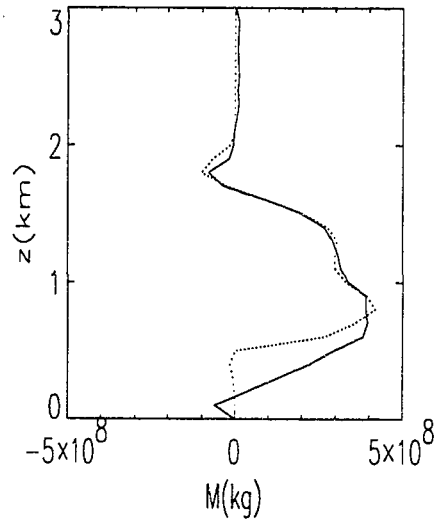

Figure 7.7: Illustration of the mass fluxes for the Hawaiian run. The solid line is M_ζ and the dotted line is M_w.

The two mass flux profiles are very similar between cloud base and cloud top. The entrainment peak in cloud base of E_w matches the integral of all the subcloud entrainment in E_ζ, so that $M_w = M_\zeta$ at cloud base. The agreement of M_w and M_ζ indicates that the mass that entered the CAR at a given level above cloud base probably originated from this level. This pattern was also seen in the other test cases to be described. It is a strong argument for lateral entrainment. There is some evidence of downdrafts driven by mixing at cloud top and evaporative cooling as seen by the negative mass fluxes that occur there.

7.3 HARP observations

We have compared mass fluxes from our model with observations of an isolated cumulus cloud that occurred on August 21, 1990 during the HARP experiment. This sounding was very similar to the sounding presented in Section 2.3 and is compared in Figure 7.8. The observed sounding (which included a very shallow layer of dense island outflow air above 50 m) was modified slightly by homogenizing all of the air between 50 and 500 meters and replacing the outflow layer below by homogenized air from above. This change removed stable stratification from below cloud base in the observed sounding which otherwise suppressed convection in our model runs. In addition to the mass flux profiles, this numerical model presented a useful comparison to the observations as well as provided a wealth of information that could not be observed.

As part of HARP, the National Center for Atmospheric Research (NCAR) CP-3 and CP-4 Doppler radars were positioned 17.5 km apart on the windward case of Hawaii. The two radars made coordinated coplanar scans that produced a dual-Doppler analysis area, or lobe, extending approximately 30 km off-shore. A dual-Doppler lobe is the region in which two radars scanned from sufficiently different perspectives to record distinct components of particle velocity. On several occasions, the NCAR Electra simultaneously flew stacked penetrations through the clouds within the Doppler area, providing comparisons of wind velocity and liquid water concentration and offering near-environmental soundings.

The radar is able to detect precipitation particles and turbulent variations in relative humidity associated with cumulus cloud by the radar scattering that they produce. From the delay and Doppler shift of the returned radar echoes, one obtains from each radar the radial velocity of the scatterers at all points in space where there is a detectable echo. Grinnell et al. (1995) were able to produce vertical mass flux profiles from the two radial velocity components by vertically integrating the mass continuity equation inside of the radar echo, assuming a zero vertical velocity at the ocean surface. Individual vertical velocities derived by this technique were fairly noisy, but the echo-integrated vertical mass flux could be determined more accurately.

For the simulation presented here, we used a 6 km by 6 km wide by 3 km deep coarse domain with a resolution of 100 m in all directions and a time step equal to 10 seconds. Nested inside this domain was a refined 2 km by 2 km wide by 2 1/2 km deep refined domain with a resolution of 33 m and a time step of 3 seconds. A Kessler-type warm rain parameterization was used to include the effect of precipitation on the convection (Kessler 1969; Chen and Cotton 1987). We initialized the cloud as a spherical bubble of air of radius 300 m, centered 300 m above the ground, by increasing q_t by 0.3 g/kg and θ_l by 0.5 K. These perturbations are similar to those observed beneath active cloudy updrafts in a trade cumulus boundary layer (Lemone and Pennell 1976). The properties of the model cloud that evolved were driven mainly by the conditional instability of the sounding and did not vary significantly with 0.1 g/kg or 0.1 K changes in the properties of the initial cloud bubble. The mass fluxes were computed over the coarse timestep according to (7.5) and (7.11). We ran the simulation for 40 minutes, but after 22 minutes the cloud ceased to evolve and the vertical velocity field became stagnant.

There is a difference between the region we define as the CAR and the one observed by the radar. The radar is only able to discern the regions where precipitation is large enough to detect or cloud edges with large refractive index variations. Especially in the early phase of the cloud lifecyle, the echo may not fully sample the CAR. In later phases, the echo also includes rain-filled air that is not part of the CAR, while some mixed but

unsaturated air in the car may be echo-free. However, Grinnell et al. show that the echo and the active part of the cloud circulation do correspond fairly well.

We selected three time periods to compare with evolutionary states defined in Grinnell et al. (1995): at 10 minutes the model cloud simulated the early phase, at 15 minutes the mature and at 20 minutes the decaying. Figure 7.9 displays the profiles of the vertical mass flux for both the simulations and observations for these time periods. Each curve represents a 10 minute average centered at the indicated time. We find it interesting that despite the difference in technique for finding mass fluxes and varying differences in sizes between the observed and modelled clouds that there is qualitative agreement between the two plots in Figure 7.9. Both clouds have a distinct early and middle stages followed by a decaying phase with negative vertical mass fluxes. We conclude that the vertical distribution of mass fluxes is a stronger function of the stage of the clouds development and than it is of the size of the cloud.

We explored the thermodynamic effect of the convection on the environment. By computing horizontal averages of the environmental soundings of $\Delta\theta_l$ and Δq_t from their initial profiles, we can compute the average vertical transport of θ_l and q_t due to the cloud. These changes are seen in Figure 7.10. In the left hand plot, the cloud acted to lower θ_l above the inversion and raise θ_l below it. The right hand plot shows the cloud moistens above the inversion and dries below it. Averaged over the 20 minute lifetime of the cloud, the vertical integral of the change in θ_l and q_t gives an estimate of the vertical flux due to the convection. Converted to energy units by the latent heat of condensation, the average moisture flux is about 40 w/m^2. Similarly, we can compute the lifetime integrated flux of θ_l. The observed flux corresponds to an increase of θ_l in the lower cloud layer and a decrease in the trade inversion, an indication of deepening of the trade-cumulus boundary layer by the turbulence within the cloud.

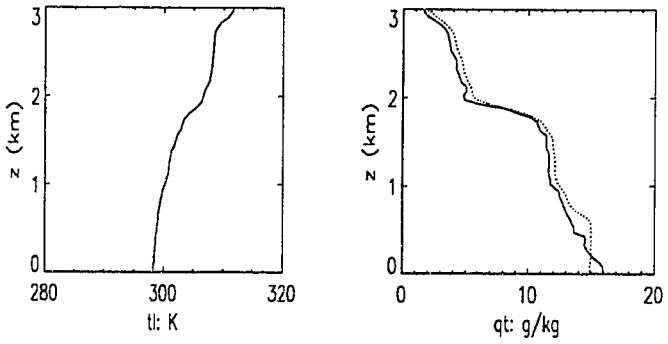

Figure 7.8: Comparison of the observed and modified soundings. The dotted line is the modified sounding and the solid is the observed.

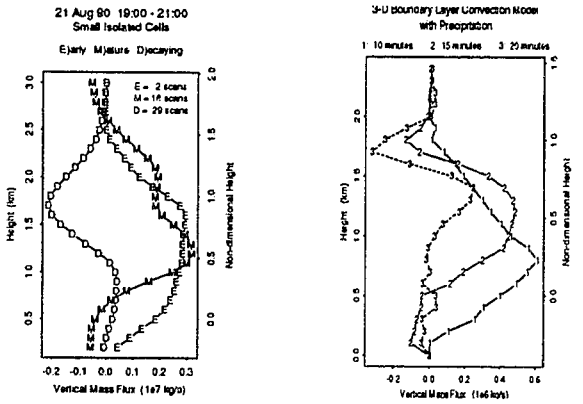

Figure 7.9: Comparison of the observed to the simulated mass fluxes

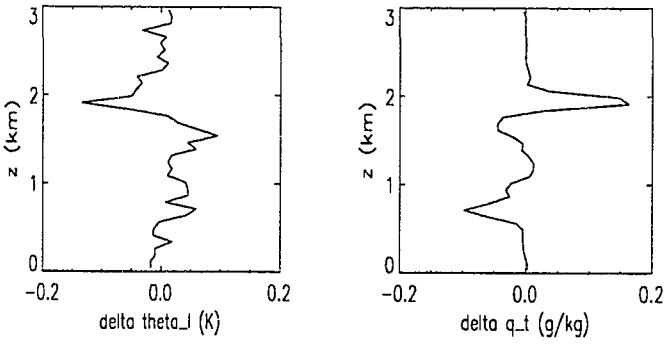

Figure 7.10: Illustration of the change in heat and moisture after $t = 20$ minutes.

7.4 Sounding Sensitivity to Shear and Relative Humidity

Numerical simulations are useful tools for assessing how changes in the thermodynamic structure of the cloud environment affect the cloud transports. We tested the effects of varying relative humidity and vertical wind shear in the cloud layer. Decreasing relative humidity increases the effect of evaporative cooling on the mixing process. Increasing the vertical wind shear affects the flow of air around and into the cloud, and tends to enhance entrainment into the cloud.

These test case used the initialization and domain of Section 7.3 but with variations of the idealized sounding in Figure 7.8. This idealized sounding, was composed of three layers, an isentropic mixed layer with fixed q_t, a cloudy layer with constant static stability and varying relative humidity and an inversion with fixed 30% relative humidity and static stability. Table 7.1 summarizes the six simulations. Simulations E0_30, E0_60, E0_80, and E0_90 test the sensitivity of cloud transports to relative humidity, while E2_80 and E4_80 show the effects of moderate and of strong vertical unidirectional wind shear.

Table 7.1 Modelled Soundings

Simulation	shear (m/(s km))	RH(%)
E0_30	0	30
E0_60	0	60
E0_80	0	80
E2_80	2	80
E4_80	4	80
E0_90	0	90

Increasing the relative humidity of the cloudy layer greatly decreases the evaporative cooling due to lateral entrainment, increasing the amount of mass that can reach the inversion (Figure 7.11). The simulation E0_80 and E0_90 are able to transport large amounts of mass to the trade inversion, while E0_30 and E0_60 are suppressed. This indicates a transition takes place between E0_60 and E0_80. The vertical mass fluxes become slightly

negative at the top of the higher humidity runs, since the updraft has enough excess buoyancy to penetrate the inversion a few hundred meters, causing it to be mixed and evaporatively cooled by this air. This feature is seen by the larger mass fluxes that are seen at the inversion of these runs.

The heat transport is strongly affected by the relative humidity of the sounding (Figure 7.12). The moist simulations EO_80 and E0_90 were able to entrain large amounts of air from above the inversion producing a strong negative flux of θ_l near the inversion. This is also seen in the average change in q_t where a large amount of moisture was lofted to the trade inversion in EO_80 and E0_90 (Figure 7.13). The drier simulations transported similar amounts of moisture upward, but to much lower heights.

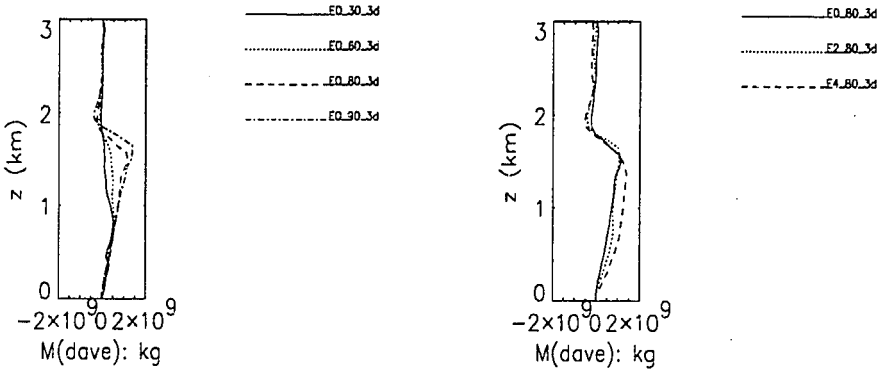

Figure 7.11: Comparison of mass flux profiles.

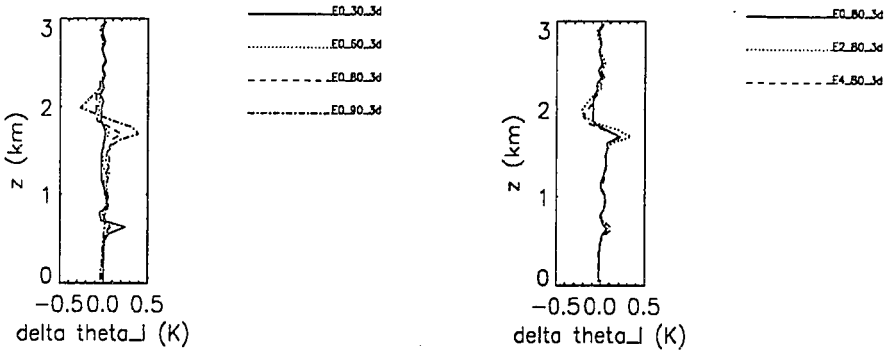

Figure 7.12: Comparison of the change in θ_l profiles.

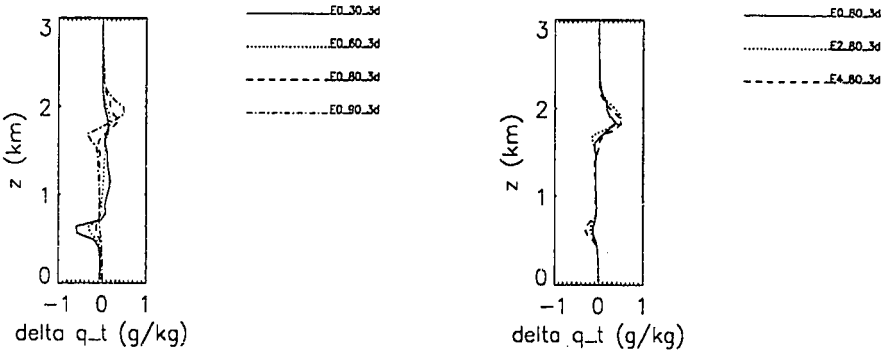

Figure 7.13: Comparison of the change in q_t profiles.

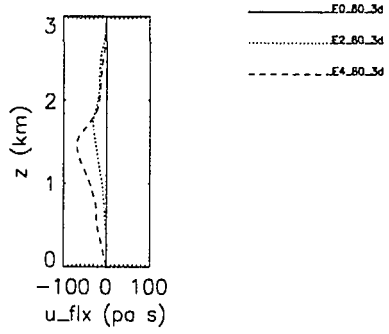

Figure 7.14: Comparison of momentum flux profiles for the cases with shear.

The effect of shear in E2_80 and E4_80 was to break up the strength of the updraft and cause the entrainment and detrainment to be redistributed downward (Figure 7.11). The shear had a small effect on the overall features of the mass flux profiles. This is an aspect of three-dimensional simulation. Mass was able to flow around the cloud undisturbed by the isolated cloud. In a two-dimensional simulation, the flow would have been blocked by the cloud and pushed upward. The heat and moisture fluxes of the sheared cases are similar to but slightly larger than the non sheared E0_80.

An important issue for the dynamics of regions with large amounts of trade cumuli is the vertical transport of horizontal momentum due to cumulus convection. The roughly linear profile of the shear is roughly reflected in the simulations E2_80 and E4_80 by negative fluxes caused by low velocity air being lifted into regions of higher velocity. From Figure 7.11 and Figure 7.14, one can estimate the average difference $u_c - u_{env}$ between u in the cloudy updrafts and u in the environment. The time integrated momentum flux over the domain area A can be estimated as $\frac{M}{A}(\overline{u'w'})$. With $M = 10^9$ kg/m^2, $A = 36\text{x}10^6$ m^2, and a time integrated momentum flux of 70 (Pa s), we find $u_c - u_{env} \approx 2\text{m/s}$, which is roughly 50% of the environmental shear between cloud base and 1.5 km. The cloud is transporting momentum fairly efficiently in the vertical.

The transport properties of the cloud are the result of the cloud transporting low level air up to the trade inversion and the compensating motions it induces in the environment. By producing an overall positive moisture flux, the cloud is able to moisten the environment more than the compensating subsidence is able to dry it out. The negative heat flux at the inversion is a result of the cloud and the subsidence working together to warm the cloud layer. This acts to destabilize the cloud layer for more and deeper convection.

These simulations support the idea of using parameterizations that favor an episodic model of cumulus convection such as the Raymond-Blyth parameterization. The simulations produce fairly uniform entrainment profiles with nonlinearly varying mass flux pro-

files. This indicates that some form of sorting is occurring where parcels mix and then proceed to be detrained at levels other than cloud top.

For this chapter, we used three dimensional domains with fixed rather than adaptive multilevel refinement. This was done for computational reasons. The presented multilevel simulations consumed over 80% of the maximum memory on our current system. It was found that peak demands placed on the system during the process of regridding the model overran the maximum amount of memory. Future work will benefit from the exponentially growing capacities of computers in terms of both memory and calculation speed.

8 Conclusions

The sharp gradients on temperature, moisture and velocities that develop around a growing cumulus cloud need to be modelled by an accurate and stable advection algorithm. This algorithm needs to be isotropic with respect to grid alignment and lack numerical dispersion and dissipation. We presented a new forward-in-time advection algorithm based on both semi-Lagrangian and Eulerian ideas. The semi-Lagrangian approach provides a means of accurately modelling flows transverse to the grid with optimal stability. Eulerian ideas were used to enforce conservation and monotonicity. This advection method was then generalized to provide a framework for solving the anelastic equations. This framework is as theoretically simple and more accurate than the traditional leapfrog schemes. It eliminates the computational problems of temporal and spatial decoupling that are associated with these methods. It minimizes memory usage by using only one time level to advance and does not have any special initialization procedures. This algorithm was then extended to become the basis of an multilevel method for the anelastic equations. We investigated numerical treatments for the conservation of mass and scalars for block composite domains and their internal interfaces.

We used this method to investigate the bulk effects of cumulus convection. It was found that the vertical mass flux profile within a small isolated shallow cumulus cloud was a function of its evolutionary stage. The evolution of the profile was in general agreement with Hawaiian cumuli derived from radar observations. Since the modelled cloud had a much smaller horizontal area than the typical Hawaiian clouds, the agreement indicates that the mass flux profile of a cumulus cloud is not strongly influenced by its size, a conclusion also found in the radar study. We explored the role of varying shear and relative humidity in the cloud layer on cumulus convection. It was found that varying relative humidity has a strong affect on the ability of the cloud to transport mass. Dry cloudy layers inhibit the convection. Shear was found to have a surprisingly weak effect on the mass fluxes of the cloud. This was hypothesized to be a three-dimensional effect. The environmental flow is deflected around the cloud without inhibiting its vertical development.

The analysis and application of multilevel methods have been hindered by the formidable programming challenges involved with implementation of these methods. Advances in computer science has recently enabled much progress in these methods to the extent that researchers will soon be able to directly compare different multilevel formulations of a given problem. By combining the use of object-oriented numerics, C++ and optimized FORTRAN, we were able to implement our multilevel method as an adaptive method by a simple process of clustering and reinitialization. Object-oriented numerics enabled us to write reusable, concise, and robust software. FORTRAN enabled us to construct a computationally efficient method. There is a growing body of object oriented software for multilevel methods that have been developed for simplifying programming tasks. Examples include: LPARX (Baden et al 1991) for communication and memory management tools for parallel computers, AMR++ (Las Alamos National Laboratory) for algorithms to cluster refined domains, and visualizations software in AVS, PVWAVE, and other graphics packages. These tools enable individual researchers to create powerful, flexible multilevel models and apply them to physical problems.

We are interested in further exploring the numerical algorithms that are the basis of adaptive multilevel methods. There is much work to be done on the elliptic equation for the pressure. A popular area of recent research in parallel computing is domain decomposition of elliptic problems. Domain decomposition is the concept of dividing the computational domain into patch grids and operating on these patches as independent objects with a fixed amount of communication. A multilevel method generalizes this by letting different patches have varying resolution and overlap each other. A natural way to parallelize elliptic solvers is to assign each of these patches to an individual processor. The available number of processors might determine the number and size of patches used. Future algorithmic advances in parallel computing will increase the performance of multilevel computations and improve their accuracy.

Many atmospheric flows involve flow around complex three-dimensional obstacles, such as the flow past the island of Hawaii. This flow has features that extend away from

the island in the form of shear lines on the northern and southern ends of the island and trailing vortices on the lee side of the island. Two popular methods for simulating these flows with grid based methods are terrain fitted grids and immersed interface methods. Immersed interface methods involve topography that dissects a Cartesian grid. The presence of topography is enforced by computing forcing terms that drive the flow around the topography. These methods are able to use fast solvers that have been created for uniform grids. However, their accuracy has not yet been fully established and they have computational inefficiencies associated with computing the dissected cells (Li and LeVeque; Almgren et al). Terrain fitted grids enjoy the benefit that they are Cartesian in computational space and thereby retain the advantages of uniform grid indexing. These methods suffer from the disadvantage that the coordinate transformed equations are not simple to integrate. A simple terrain fitted grid that has been used extensively for atmospheric models is the vertical transformation

$$\bar{z} = \frac{z - z_s}{H - z_s} \tag{8.1}$$

where z_s is the height of the terrain (Clark 1977; many others). This transformation provides a way of extending our research of multilevel methods.

The multilevel implementation of meteorological processes presents considerable numerical challenges. The parameterization of rain and ice involve convective fluxes due to the differential fall speeds involving rain droplets and ice particles. This poses important conservation issues, if we are to retain consistent heat and moisture budgets across internal boundaries. Radiation is a nonlocal process that occurs throughout the depth of the domain. This process often predicts sharp sources and sinks that could benefit from a multilevel treatment. Radiation is computed implicitly via computing optical depths through layers of cloud. A multilevel treatment offers the promise of improved accuracy in radiation calculations by providing finer increments through which to compute these optical depths.

We intend to further our work in applying adaptive multilevel ideas to the numerical simulation of boundary layer convection. Future research will involve stratocumulus convection. Stratocumulus convection involves radiation and other forcings. An example of this is the multilevel simulation of Figure 8.1 where two stages in the evolution of a stratocumulus cloud are shown. The cloud was initialized as a calm isotropic region with uniform total moisture below 700 meters that has condensation between 300 and 700 meters. At 700 meters there is a strong inversion characterized by a sharp increase in temperature and decrease in moisture. Above the inversion, the environment is stably stratified with constant total moisture. This simulation is typical of a nighttime stratus cloud with radiative cooling at the top of the cloud and warming at the bottom which drives the resultant eddies. Figure 8.1 shows the q_c field at $t = 4$ minutes, when the eddies are in their initial development and at $t = 50$ minutes when they have fully formed.

This simulation involved a 1.2 km wide by 1.2 km high base domain with a resolution of 24 m and timestep of 4 seconds and an inner domain with a spatial resolution of 8 m and a timestep of 1 1/3 seconds. This simulation used a simple multilevel treatment of radiation that interpolated radiative fluxes from the coarse to the fine domain and was computed every two coarse timesteps. These are reasonable assumptions considering that radiation primarily influences motions on a timescale involving minutes. This simulation shows the merit of using a multilevel method for stratocumulus entrainment. Modelling the dynamics that occur at the stratocumulus inversion is a challenging numerical problem. The motions at the inversion are difficult to model due to the sharp gradients in heat and moisture that are diffused in unrefined numerical models. The entrainment and mixing of environmental air at the inversion has an important effect on the dynamics of the cloud. This mixing and entrainment creates regions of evaporative cooling that drive eddies down into the cloud. This is seen in the bottom figure of Figure 8.1, where on the right of the domain a tongue of dry environmental air has been pulled into the cloud.

We plan to use our adaptive multilevel model to simulate an interacting field of cumulus clouds evolving naturally from a turbulent subcloud layer. This allows us to improve

on work of Sommeria (1978) to produce accurately budgets for the trade wind cumulus layer and insights into cloud organization. These clouds arise as small eddies that combine into larger ones due to environmental forcings and are in constant transition. An example of this is seen in Figure 8.2, where a simulation of an evolving cumulus layer is given. The model was initialized with the sounding in Section 2.3 with a constant surface flux at the bottom of the domain of 200 W/m^2 heat and 200 W/m^2 moisture. The criteria for marking cells was the presence of liquid water. This figure illustrates the utility of a multilevel method to increase the efficiency and accuracy of a model.

Larger convective systems can also be effectively simulated with an adaptive method (Skamarock et al 1993). Large regions of the tropical oceans have little or no convection with isolated regions of deep convection (mesoscale convective systems) embedded within them. Mesoscale convective systems interact to form superclusters and large-scale tropical waves that dominate weather and climate in the equatorial belt. Many other problems in geophysical fluid mechanics involve a good understanding of the interaction between motions of widely disparate length scales in a small but changing part of the flow. We anticipate that adaptive multilevel methods will become a widely used and powerful tool for simulating and understanding such flows.

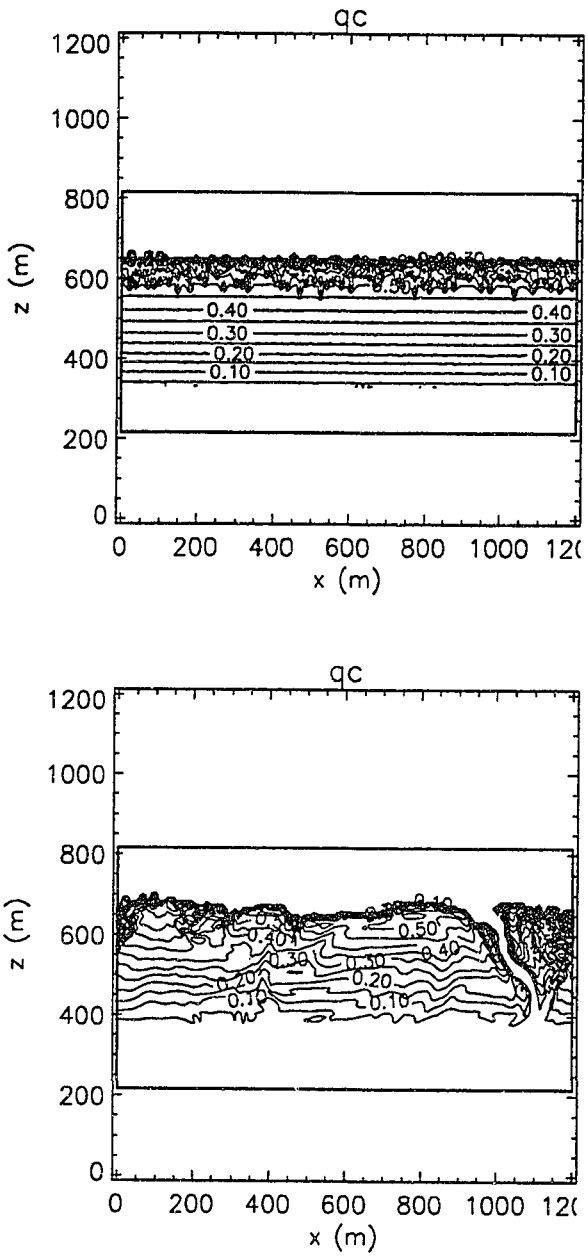

Figure 8.1: Illustration of radiatively cooled stratocumulus cloud at t = 4 minutes (top) and t = 50 minutes (bottom). Contours are in g/kg with each contour representing 0.05 g/kg increments.

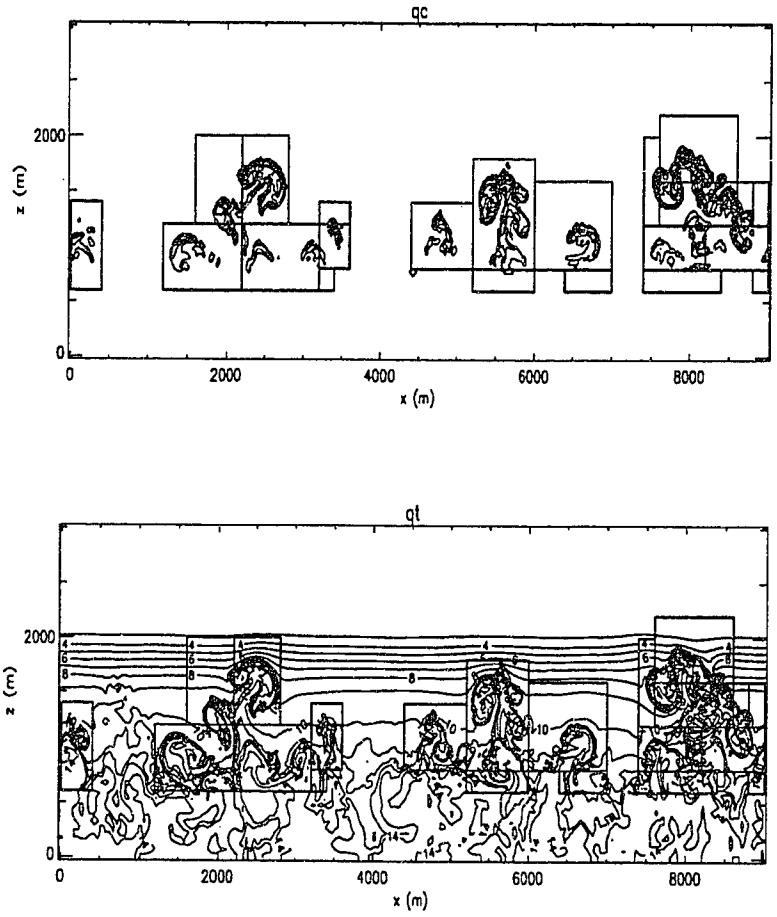

Figure 8.2: Illustration of an actively convecting boundary layer.

Bibliography

Almgren, A. S., J. B. Bell, P. Collela, and L. H. Howell, 1993: An Adaptive Projection Method for the Incompressible Euler Equations. Lawrence Livermore National Laboratory UCRL-JC-113853 Preprint.

Arakawa, A., 1966: Comutational design for long term integration of the equations of motion: Two-dimensional incompressible flow. Part I. *J. Comput. Phys.* **1**, 119-143.

Arakawa, A., and W. H. Schubert, 1974: Interaction of a Cumulus Cloud Ensemble with the Large Scale Environment, Part. 1. *J. Atmos. Sci.*, **31**, 674-701.

Asselin, R. A., 1972: Frequency filter for time integrations. *Mon. Wea. Rev.*, **100**, 487-490.

Baden, S. B., S. R. Kohn, and S. J. Fink, 1991: Programming with LPARX. CSE Technical Report Number CS91-377.

Beheng, K. D., 1994: A parameterization of warm cloud microphysical conversion processes. *Atmospheric Research*, **34**, Part II, 193-206.

Bell, J. P., and D. L. Marcus, 1989: A Second-Order Projection Method for Variable-Density Flows. *J. Comput. Phys.*, **100**, 334-348.

Berger, M., and J. Oliger, 1984: *J. Comput. Phys.* **53**, 484.

Berger, M., 1987: On Conservation at Grid Interfaces. *SIAM J. Numer. Anal.* **24**, 967-984.

Berger, M., and P Colella, 1989: Local Adaptive Mesh Refinement for Shock Hydrodynamics. *J. Comput. Phys.* **82**, 64-84.

Berger, M., and I. Rigoutsos, 1991: An Algorithm for Point Clustering and Grid Generation. *IEEE Transactions on Systems, Man and Cybernetics.* **21**, 1278-86.

Bretherton, C. S., and P. K. Smolarkiewicz, 1989: Gravity waves, compensating subsidence and detrainment around cumulus clouds. *J. Atmos. Sci.*, **46**, 740-759.

Budd, T., 1991: An introduction to object-oriented programming, Addison-Wesley Publishing Company.

Chen, C., and W. R. Cotton, 1987: The physics of the marine stratocumulus-capped mixed layer. *J. Atmos. Sci.*, **44**, 2951-2977.

Chorin, A. J., 1967: *J. Comput Phys.* **2**, 12.

Chorin, A. J., 1969: On the convergence of discrete approximations to the Navier-Stokes Equations. *Math. Comp.*, **23**, 341-353.

Clark, T. L. , 1977: A small-scale dynamic model using a terrain-following coordinate transformation. *J. Comput. Phys.*, **24**, 186-215.

Clark, T. L., and W. R. Farley, 1984: Severe downslope windstorm calculations in two and three spatial dimensions using anelastic grid nesting: A possible mechanism for gustiness. *J. Atmos. Sci.*, **41**, 329-350.

Durran, D. R., and J. B. Klemp 1983: A compressible model for the simulatioin of moist mountain waves. *J. Atmos. Sci.*, **111**, 2341-2361.

Durran, D. R., 1989: Improving the anelastic continuity equation. *J. Atmos. Sci.*, **46**, 1453-1461.

Godunov, S. K., 1959: *Mat. Sb.*, **47**, 271.

Grabowski, W. W., and T. L. Clark, 1991: Cloud-environment interface instability: Rising thermal calculations in two spatial dimensions. *J. Atmos. Sci.*, **48**, 527-546.

Harlow, F. H., and J. E. Welch, 1965: Numerical calculation of time-dependant viscous incompressible flow of fluid with free surface. *Phys. Fluids*, **8**, 2182-2189.

Kessler, E., 1969: On the distribution and continuity of water substance in atmospheric circulations. *Met. Monograph, Vol. 10, No 32*, American Meteorological Society, Boston.

Klaassen, G. P., and T. L. Clark, 1985: Dynamics of the cloud-environment interface and entrainment in small cumuli: Two-dimensional simulations in the absence of ambient shear. *J. Atmos. Sci.*, **42**, 2621-2642.

Klemp, J. B., and R. B. Wilhelmson, 1978: The simulation of three-dimensional convective storm dynamics. *J. Atmos. Sci.*, **35**, 1070-1095.

Klemp, J. B., and D. R. Durran, 1983: An upper boundary condition permitting internal gravity wave radiation in numerical mesoscale models. *Mon. Wea. Rev.*, **111**, 430-444.

Lemone, M. A., and W. T. Pennell, 1976: Relationship of trade wind cumulus distributions to subcloud layer fluxes and structure, *Monthly Weather Review*, **104**, 524-539.

Leonard, B. P., M. K. MacVean, and A. P. Loch, 1993: Positivity preserving numerical schemes for multidimensional advection. NASA Technical Memorandum 1060555, ICOMP-93-05, NASA Lewis.

LeVeque, R. J., 1993: High-resolution Algorithms for Advection in Incompressible Flow. *SIAM J. Numer. Anal.*, submitted December, 1993.

LightHill, J., 1978: Waves in Fluids, Cambridge University Press.

Lippmann, S. B., 1993: C++ Primer, 2nd edition, Addison-Wesley Publishing Company.

Lipps, F. B. and R. S. Hemler, 1982: A scale analysis of deep moist convections and some related numerical calculations. *J. Atmos. Sci.*, **39**, 2192-2210.

Ogura, Y. and N. Phillips, 1962: Scale analysis of deep and shallow convection in the atmosphere. *J. Atmos. Sci.*, **19**, 173-179.

McCormick, S. F., 1989: Multilevel Adaptive Methods for Partial Differential Equations. *Frontiers in Applied Mathematics*, SIAM.

Nicholls, M. E., R. A. Pielke, and W. R. Cotton, 1991: Thermally force gravity waves in an atmosphere at rest. *J. Atmos. Sci.*, **48**, 1869-1884.

Paluch, I. R., 1979: The entrainment mechanism in Colorado cumuli. *J. Atmos. Sci.*, **36**, 2467-2478.

Raga, G. B., J. B. Jensen and M. B. Baker, 1990: Characteristics of cumulus band clouds off the coast of Hawaii. *J. Atmos. Sci.*, **47**, 338-355.

Rasmussen, R. M. , P. K. Smolarkiewicz and J. Warner, 1989: On the dynamics of Hawaiian cloud bands: Comparison with observations and island climatology. *J. Atmos. Sci.*, **46**, 1589-1608.

Raymond, D. J., and A. Blyth, 1986: A stochastic mixing model for nonprecipitating cumulus clouds. *J. Atmos. Sci.*, **43**, 2708-2718.

Raymond, D. J., R. Solomon and A. M. Blyth, 1991: Mass fluxes in New Mexico mountain thunderstorms from radar and aircraft measurements. *Quart. J. Roy. Meteor. Soc.*, **117**, 587-621.

Robert, A. J., 1966: The integration of a low order spectral form of the primitive meteorological equations. *J. Meteor. Soc. Japan*, **44**, 237-245.

Shih, T. M., C. H. Tan and B. C. Hwang, 1989: Effects of grid staggereing on numerical schemes, *Int. J. Numer. Methods Fluids*, **9**, 193-212

Siems, S. T. and C. S. Bretherton, 1992: A numerical investigation of cloud-top entrainment instability and related experiments. *Quart. J. Roy. Meteor. Soc.*, **118**, 787-818.

Skamarock, W. C. and J. B. Klemp, 1992: The Stability of Time-Split Numerical Methods for the hydrostatic and nonhydrostatic elastics equations, *Monthly Weather Review*, **120**, 2109-2127.

Skamarock, W. C. and J. B. Klemp, 1993: Adaptive grid refinement for two-dimensional and three-dimensional nonhydrostatic atmospheric flow, *Monthly Weather Review*, **121**, 788-804.

Skamarock, W. C. and J. B. Klemp, 1994: Efficiency and accuracy of the Klemp-Wilhelmson time-splitting technique, *Monthly Weather Review*, **122**, 2623-2630.

Skamarock, W. C., M. L. Weisman, and J. B. Klemp, 1994: Three-dimensional evolution of simulated long-lived squall lines. *J. Atmos. Sci.*, accepted.

Sommeria, G., 1975: Three-dimensional simulation of turbulent processes in an undisturbed trade wind boundary layer. *J. Atmos. Sci*, **33**, 216-241.

Smolarkiewicz, P. K., 1984. A fully multidimensional positive definite advection transport algorithm with small implicit diffusion. *J. Comp. Phys.*, **54**, 325-362.

Smolarkiewicz, P. K., and T. L. Clark, 1985: Numerical simulation of the evolution of a three-dimensional field of cumulus clouds. Part I: Model description, comparison with observations and sensitivity studies. *J. Atmos. Sci.*, **42**, 502-522.

Smolarkiewicz, P. K., and T. L. Clark, 1986. The multidimensional positive definite advection transport algorithm: Further development and applications. *J. Comp. Phys.*, **67**, 396-438.

Smolarkiewicz, P. K., R. M. Rasmussen and T. L. Clark, 1987: On the dynamics of Hawaiian cloud bands: Island forcing. *J. Atmos. Sci.*, **45**, 1872-1905.

Smolarkiewicz, P. K., and W. W. Grabowski, 1989: The multidimensional positive-definite advection transport algorithm: nonoscillatory option. *J. Comp. Phys.*, **86**, 355-375.

Smolarkiewicz, P. K., and J. A. Pudykiewicz, 1992: A class of semi-lagrangian approximations for fluids. *J. Atmos. Sci.*, **49**, 2082-2096.

Smolarkiewicz, P. K., and L. G. Margolin, 1993: On forward-in-time differencing for fluids: extenstion to a curvilinear framework. *Monthly Weather Review.*, **121**, 1847-59.

Strang, G., 1968: On the construction and comparison of difference schemes. *SIAM J. Num. Anal.*, **5**, 506-517.

Telford, J. S., 1975: Turbulence, entrainment and mixing in cloud dynamics. *PureAppl. Geophys.*, **113**, 1067-1084.

Tripoli and Cotton, 1980: *J. Appl. Met.*, **19**, 1040-1041.

Vermeulan, A., 1992: Object oriented numerics, *Preprint*, 1-16.

Vermeulen, A., 1993: Object-oriented finite-element software, *Dr. Dobb's J.* **18**, 86-91.

Wilhelmson, R. B., and J. H. Erickson, 1977: Direct solutions for Poisson's equation in three dimensions, *J. Comput. Phys.*, **25**, 313-331.

Wong, M. K. W., K. G. Budge, J. S. Peery, and A. C. Robinson, 1993: Object-oriented numerics: A paradigm for numerical object-oreiented programming, *Comp. Phys.* **7**, 655-663.

Zalesak, S. T., 1979: Fully multidimensional flux corrected transport algorithm for fluids, *J. Comput. Phys.*, **31**, 335-363.

Zang, Y., R.L. Street and J. R. Koseff: 1994: A non-staggered grid, fractional step method for time-dependant incompressible Navier-Stokes equations in Curvilinear Coordinates, *J. Comput. Phys.*, **114**, 18-33.

VITA

David Eric Stevens

Born: February 15, 1966; Denver, Colorado.

Parents: Clarence O. and Marcia K. Stevens.

Education:

Master of Science.
Applied Mathematics.
University of Washington, Seattle, Washington.
June, 1991.

Bachelor of Science
Applied Mathematics, minor in Chemical Engineering.
University of Colorado, Boulder, Colorado.
May, 1988.

Thomas Jefferson High school, Denver, Colorado.
May, 1984.