UNIVERSITY OF NAIROBI

AN INVESTIGATION OF NUMERICAL SOLUTION TO PARTIAL DIFFERENTIAL EQUATIONS IN NUMERICAL WEATHER PREDICTION

ANGWENYI NYACHAE DAVID

I56/74362/2012

SUPERVISOR

DR. NYANDWI CHARLES

A Dissertation in partial fulfilment for a Master of Science degree in Applied Mathematics

NAIROBI

AN INVESTIGATION OF NUMERICAL SOLUTION TO PARTIAL DIFFERENTIAL EQUATIONS IN NUMERICAL WEATHER PREDICTION

Angwenyi Nyachae David

23rd June, 2013

Declaration

I, the undersigned, declare that this project report is my original work, and, to the best of my knowledge, has not been presented for award of a degree in any other university.

Signature : _____

Name : ANGWENYI NYACHAE DAVID

Reg. No : I56/74362/2012

Date : _____

Declaration by Supervisor

This project report has been submitted for examination with my approval as supervisor.

Signature : _____

Name : DR. NYANDWI CHARLES

Date : _____

Statement

This dissertation has been submitted in partial fulfilment of requirements for a Master of Science degree at The University of Nairobi and is deposited in the University Library to be made available to borrowers under the rules of the Library.

Acknowledgements

My first and foremost tribute goes to our God, the only dependable Father, and Christ Jesus, the only Potentate and Bishop of our souls, from whom every good and perfect gift comes from, and in whom we live, move and have our being. To Him, I am grossly and eternally indebted in all things, including those things that pertain to life and godliness.

The debt of gratitude I owe Masinde Muliro University of Science and Technology for the nicest scholarship they proffered me cannot be quantified. The fee they paid ensured my full stay in class and the substantial stipend they promptly conferred me spared me of many an affliction that could be my lot had it been otherwise.

For this project and for similar studies I have been privileged to obtain, I thank the University of Nairobi, especially the School of Mathematics. It has been my academic home. The ablest Lecturers of this esteemed university, under whose tutelage I have happily been, deserve my deepest appreciation.

My finest acknowledgement goes to Dr. Charles Nyandwi, my supervisor in this project, for his invaluable counsel and friendly presence in the course of executing this work. The introduction he gave me to Partial Differential Equations is gratefully acknowledged and should not pass unmentioned. I equally thank Prof. Ogana for the basis he laid for us in Numerical Solution to Partial Differential Equations, whose use in this project will remain to attest to it.

With most joyful emotion, I thank our immediate family members, especially my mother, Mrs Angwenyi Peris, my sisters, Caroline and Susan, and my beloved brother, Robert, for their fond affection, ardent prayers and best wishes for me and my affairs even when I was far from them. The afflictions they have borne for want of my company are immensely acknowledged. I would like to pay special tribute to Uncle Kennedy who ensured my registration for Masters.

What more shall I write but to thank my worthy friends and classmates; Pepela, Onchomba, Masibayi and Kipchumba for the sweet company in the pursuit of knowledge. I single out Mr. Andrew Masibayi for being a close ally and room-mate. His assistance and presence contributed substantially to the blessedness of this course.

“For those whose love for Mathematics supersede their love for food”

Table of Contents

List of Tables

Table of Graphs

Table of Figures

Definition of key terms, concepts and variables

1. **Weather forecasting** entails predicting how the present state of the atmosphere will change. Present weather conditions are obtained by ground observations, observations from ships and aircraft, radiosondes, Doppler radar, and satellites. This information is sent to meteorological centers where the data are collected, analyzed, and made into a variety of charts, maps, and graphs. These charts, maps, and graphs are then sent electronically to forecast offices where local and regional weather forecasts are made [13].

2. **The atmosphere** is a mixture of gases surrounding any celestial object (in this case the Earth) that has a gravitational field strong enough to prevent the gases from escaping.

3. **Centrifugal force** is the apparent outward force that draws a rotating body away from the centre of rotation. It is caused by inertia of the body as the body's path is continually redirected.

4. **Coriolis force** is additional force acting on the motion of bodies in a rotating system of reference. Once the air has been motioned by the pressure gradient force, it undergoes an apparent deflection from its path, as seen by an observer on the earth. This apparent deflection because of the Earth's rotation is called Coriolis force.

5. **Advection** is the horizontal transfer of a property such a heat caused by air movement.

6. **Troposphere** is the lowest layer of the earth's atmosphere and site of all weather on earth. The troposphere is bounded on the top by a layer of air called the tropopause, which separates the troposphere from the stratosphere, and on the bottom by the surface of the Earth. The troposphere is wider at the equator (16km/10mi) than at the poles (8km/5mi).

7. **Pressure** is the force per unit area exerted by a liquid or a gas on a body or a surface, with the force acting at right angles to the surface uniformly in all directions.

8**. Humidity** is the moisture content of the atmosphere. The atmosphere always contains some moisture in the form of water vapour; the maximum amount depends on the temperature.

9. **Temperature** is the property of systems that determines whether they are in thermal equilibrium. The concept of temperature stems from the idea of measuring relative hotness and coldness and from the observation that the addition of heat to a body leads to an increase in temperature as long as no melting or boiling occurs.

10. **Wind** is air in motion. The term is usually applied to the natural horizontal motion of the atmosphere; motion in a vertical or near vertical direction is called a current. In this work, components of wind are considered in the zonal, meridional and radial directions.

11. **Density** is the ratio of the mass of a substance to its density.

12. **Numerical Weather Prediction** is a method of weather forecasting that employs a set of equations that describe the flow of fluids, which is translated into computer codes, combined

with parameterization of the processes that cannot be adequately formulated on a specific domain, and integrated based on the initial and boundary conditions.

Abstract

In this study, we begin by presenting an overview of the Numerical Weather Prediction process as used in the Unified Model of the Met Office in UK. The primitive equations are the continuity equation, the momentum equations, equations for representation of moisture, the expression for the first law of thermodynamics and the equation of state. Discretisation of these equations is done using the two-time-level, off-centred, Semi-Implicit, Semi-Lagrangian time discretisation scheme. This is preferred to the Eulerian decomposition in which advection terms abound, rendering it computationally inefficient. Consistency and stability of this scheme is analysed; the latter using the matrix method of stability analysis. Convergence of the SISL scheme is inferred by using Lax Equivalence Theorem.

The coupling of the discretised governing equations results in the Helmholtz equation whose solution yields the increment in pressure field, Π'. To analyse the condition for stability of the Helmholtz equation we have used the Von Neumann approach, which shows that the spatial-steps chosen and the wave number are factors that affect stability. In the final analysis, the recurrence relation for a $2-D$ Helmholtz equation is solved using Jacobi, Gauss-seidel, Successive-Over-Relaxation, Conjugate Gradient, Bi-Conjugate Gradient, Bi-Conjugate Gradient Stabilized, Quasi-Minimal Residual and Gradient Minimal Residual methods. Respective iteration time is also shown. We show that Bi-CGSTAB method is most efficient, followed by GMRES method. Finally, a visual aid for the solution of 2-dimensional Helmholtz equation is shown.

CHAPTER 1: INTRODUCTION

1.1 Background of the study

The earliest works on Numerical Weather Prediction can be traced back to the contributions of the great American meteorologist Cleveland Abbe in 1890. He recognized that "meteorology is essentially the application of hydrodynamics and thermodynamics to the atmosphere." He proposed a mathematical approach to forecasting. He was optimistic that the scientists of the atmosphere would "take up our problems in earnest and devise either graphical, analytical or numerical methods" of solving equations [5].

This was taken up by Vilhelm Bjerkness who used the diagnostic and prognostic steps to predict weather. However, he was confronted with the challenge of absence of adequate information about weather over seas and in the upper air. He identified the following prognostic variables: pressure, temperature, humidity, density and the three components of velocity. The primitive equations he used are the three hydrodynamic equations of motion, the continuity equation, the equation of state and the equation expressing the first and the second laws of thermodynamics. Eliassen (1999) however, pointed out that Bjerkness was in error in listing the second law of hydrodynamics; he should have instead specified the continuity equation for water substance. He used the qualitative graphical methods for solving the partial differential equations that he obtained.

Lewis Fry Richardson attempted a direct solution of equations of motion. He remarked that the atmosphere is complicated and therefore the scheme for weather forecasting is equally complicated. Nonetheless, he was optimistic saying that: "perhaps someday in the dim future it will be possible to advance the computations faster than weather advances at a cost less than the saving to mankind due to information gained." The basic equations that had been identified by Abbe and Bjerknes were simplified by hydrostatic assumption and he rendered them amenable to approximate solution through transformation. He preferred the numerical method of solving the differential equations. Longitude, latitudes and height formed grid points. Unfortunately, his first worked problem did not yield realistic results because of the complications in initialization.

The advances made by Richardson attracted the interest of one of the leading mathematician of the 20th century, John Von Neumann. He noticed that versatile computing machinery was inevitable for solving the complex equations of turbulent flow. He rallied behind the construction of an electronic computer in the Institute of Advanced Technology in Princeton. The successful implementation of this project became a milestone in Numerical Weather Prediction.

Later development saw the simulation of realistic weather forecasts. "Using the simplified equations proposed by Jule Charney (known as the barotropic vorticity equation) the first successful forecast was produced with ENIAC (Electronic Numerical Integrator and Computer) in April, 1950 for a level in the middle troposphere (500hpa) supposed to represent the total behaviour of the atmosphere." [5] This forms the basis of Numerical

Weather Prediction. It is surprising that a forecast of 24 hours took 36 hours of the computer time!

To alleviate this challenge, supercomputers that are more versatile were developed. Even so, the efficiency of the prediction process has been a challenge. There is a need to modify the models by simulating the weather using efficient numerical methods that converge to a solution faster than the existing ones.

Although few mathematicians have taken interest in Numerical Weather Prediction, it largely remains the province of meteorologists. This should not be so. Ingenious numerical methods developed by mathematicians could help in attaining efficiency and optimality of weather forecasts. That very few books with mathematical treatment of weather prediction have been written attests to this need.

This study, therefore, comes at the right time. Whereas it is intended to investigate the numerical solution of partial differential equations that form part of Numerical Weather Prediction already in place, attempts will be made to consider and suggest other numerical solutions that are relatively stable and that converge easily to the intended solution.

1.2 Problem description/Research problem

1.2.1 Background of the problem

The laws of nature are fashioned in the language of partial differential equations. Evolution of weather, just like many physical phenomena, can be described using differential equations. Since weather is the condition of the atmosphere for a short period, any attempt to study weather involves a close study of the atmosphere.

The atmosphere is a fluid. To study weather therefore, the equations governing the flow of fluids are inevitable. These equations are mainly seven: the three hydrodynamic equations of motion, the continuity equation, the equation of state and the equations expressing the first and second laws of thermodynamics (Peter Lynch 2006). Seven basic variables are of interest: pressure, temperature, density, humidity and the three components of velocity *(ibid)*.

In the past, graphical solution of equations was used in Weather Prediction. Forecasts were done by comparing different charts. It was based on intuition and intelligent guess. Analytical solutions to the partial differential equations that describe weather are cumbersome. Some partial differential equations that arise in weather prediction do not have analytical solutions.

Use of numerical methods in weather prediction has been in the rise. Iterative methods are preferred that converge to a solution fast and that do not require a lot of storage. In many

cases, preconditioners are used to accelerate convergence. However, introduction of preconditioners come with the additional cost of storage of the preconditioner itself.

The major bottleneck in Numerical Weather Prediction is efficiency and optimality of the iterative methods used. In Met Office, for example, solving of the Helmholtz equation takes 35% of the total forecast time! A search is on for an efficient method.

We model the weather on a sphere. Ideally, the earth is oblate spheroidal. This is attributable to the interaction of centrifugal and gravitation forces (Staniforth *et. al.* 2004). In the expression for the equation of conservation of momentum the apparent vertical / apparent horizontal decomposition is necessary in order to separate the Coriolis force from the centrifugal force; but the Newtonian gravity dominates over centrifugal effects that geopotentials may be represented as (concentric) spheres to a very good approximation (*ibid*).

The English Quaker scientist Lewis Fry Richardson expressed a hope that "perhaps someday in the dim future it will be possible to advance the computations faster than the weather advances…" (Richardson 1922). Overtime, mathematicians and meteorologists have been trying to fulfil this hope. There has been a need to obtain efficient solutions for weather simulation. Great steps have been made, thanks to supercomputing. It is the bounden duty of this study to pursue the realization of this dream.

1.2.2 Statement of the problem

We are studying the solution to the partial differential equations in Numerical Weather Prediction because we want to find out how these solutions have been obtained and their efficiency in order to help readers understand how meteorologists have solved these equations so that the readers can use the most efficient solution methods.

1.3 Objective of the study

This study is of practical contribution. It is intended to investigate efficient numerical solutions to the partial differential equations that arise in simulation of weather. The findings will help in the Numerical Weather Prediction models that are in existence to give forecasts optimally and over a short period.

1.3.1 Specific objectives

- To study the numerical solution of partial differential equations in Numerical Weather Prediction
- To analyse consistency, convergence and stability of the scheme used in Numerical Weather Prediction.
- To find the most efficient numerical solution for weather simulation

1.3.2 Research question

This study is made because:

- We want to find out how weather is simulated
- We want to find out whether there is a most efficient numerical solution to the equations in Numerical Weather Prediction.
- We want to analyse stability, consistency and convergence of the scheme used in Numerical Weather Prediction.

1.3.3 Limitation of study

As is expected this study could have used the current available data for analysis. This has not been done. However, a sample problem has been used for the purposes of giving an overview of what is happening in the real cases. This is attributed to the costs that are incurred when procuring data and the complexity of the real life problems given the limited research time. In spite of this, it is our ardent hope that the simple sample problem used will illustrate the principles that can be seen in global scale. We suggest also that in future work consideration will be given to practical implementation of the ideas developed in this study.

Had means afforded, more information could be sought at Met Office, UK. A visit to the weather station would be of invaluable benefit to this study. Information on initialization, computation of forecast variables and final stages of forecasting are rare. Perhaps at the forecast site this information could be obtained.

CHAPTER 2: LITERATURE REVIEW

The model adopted for this study is the Unified Model of Met Office in UK. This model was chosen because it is recognised as the world's leader in Numerical Weather Prediction [12]. The treatment of the model from the primitive equations to the forecast stage has been shown in Staniforth *et al* (2004).

The primitive equations that form the basis of Numerical Weather Prediction have been shown to be momentum equations, continuity equation, the equation of state, the representation of moisture and the expression of the first law of thermodynamics. The primitive equations have been cast into forms amenable for practical implementation in Cullen *et al* (2005). This includes factoring in the curvature of the domain, parameterization of physical processes and expunging from the equations processes with negligible effects, whose presence could render these equations cumbersome to compute.

The governing equations are discretised using the Semi-Implicit Semi-Lagrangian time discretisation. This has been shown in Staniforth *et al* (2004), beginning from a thorough explanation of the method. It also includes a series of predictor –corrector steps for those quantities whose discretisation is intractable. The strengths of the Semi-Implicit Semi-Lagranian discretisation over Eulerian decomposition have been shown. This includes its stability even when long time steps are used and the absence of Eulerian advection terms. The treatment at the poles has been elaborated in depth.

A linear stability analysis of the two-time-level semi-implicit discretisation of the adiabatic compressible equations has been given in Payne (2008). Previous works had shown that the scheme is stable with respect to perturbations to a hydrostatic and isothermal basic state if the time-implicit weight is used throughout and is greater than $\frac{1}{2}$. This result has been generalised in Payne (2008) to the case where different time weights are used for different terms. The matrix method is used with the normal modes of the perturbed static states of the governing equations in rectangular coordinates. It was established that the scheme is stable when the time-weights are greater than $\frac{1}{2}$ and when $\alpha_3 \geq \alpha_1 > \frac{1}{2}$ [3]. However, to the best of our knowledge, consistency and convergence of the Semi-Implicit Semi-Lagrangian discretisation scheme has not been done.

Esterhazy S. and Melenk J. M. (2011) studied the stability of discretisation of the Helmholtz equation at large wave-numbers. In as much as this has been done, we have not encountered a stability analysis for the Helmholtz equation using the Von Neumann approach, in spite of its simplicity.

Whereas the Helmholtz equation has been solved both numerically and analytically, an investigation of the most efficient method among the existing iterative methods has not been shown to the best of our knowledge. In staniforth *et al* (2004) preference has been given to the LU decomposition preconditioned with Generalised Conjugate Residual method and the

Alternating Direction Implicit method. Even so, efficiency and optimality of the solution of the Helmholtz equation has been a major bottleneck.

This study is intended to highlight on the previous works done and to contribute to some missing links in these works.

CHAPTER 3: RESEARCH METHODOLOGY

The following is a flow chart that summarises the basic steps in Numerical Weather Prediction, whose overview has been shown in this study.

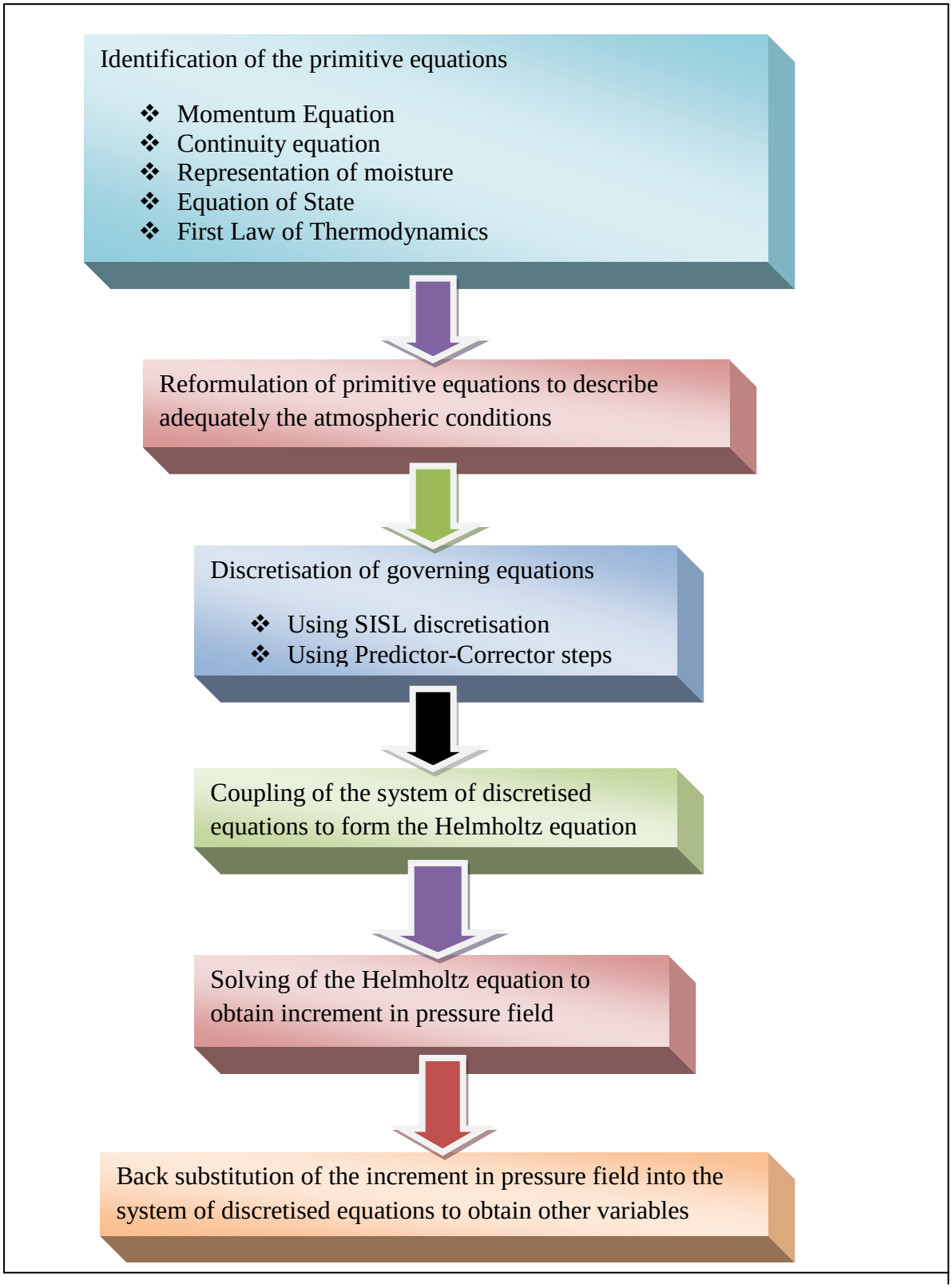


Figure 1: A flow chart showing basic steps in Numerical Weather Prediction

Figure 1 shows the basic steps as have been highlighted in this study and as are used in Numerical Weather Prediction.

We proceed to highlight on the consistency, stability and convergence of the Semi-Implicit Semi-Lagrangian time discretisation. Much of the work on stability analysis is borrowed from Payne (2008) almost entirely, excepting the examples given to illustrate stability. This is done in Chapter 5.

The analysis of stability using Von Neumann approach is done in Chapter 6. In the same chapter, a sample simple example for the Helmholtz equation is solved numerically using Jacobi Iterative method, Gauss-Seidel Iterative method, Successive-Over-Relaxation method, Conjugate Gradient Method, Bi-Conjugate Gradient method, Bi-Conjugate Gradient Stabilised method, Quasi-Minimal Residual method and Generalised Minimal Residual method. The iterations are solved using MATLAB.

The number of iteration to 4 decimal places is taken for every method and the equivalent CPU-time for the respective method is taken. The results of the MATLAB output are tabulated to a point of convergence.

In Chapter 7, the analysis of the results obtained is done. A summary table for the number of iterations for the 8 methods and their respective CPU-time is drawn. A graph of number of iterations and CPU-time for every method used is drawn.

In order to have a visual impression of the Helmholtz equation, Graphical User Interface in MATLAB is used. The figure is given in Chapter 7.

Finally, the summary of the results obtained is provided in Chapter 8. Recommendations and findings are also highlighted.

CHAPTER 4: OVERVIEW OF NUMERICAL WEATHER PREDICTION

4.1. Governing equations

This study begins with an exploration of the governing equations. Modifications of these equations are done to render them amenable for describing weather conditions on the Earth's surface. In these equations, $\frac{D}{Dt}$ stands for the material (or Lagrangian derivative); u, v, w are the three wind components; λ, ϕ, r stand for longitude, latitude and radius coordinates respectively. S stands for the tendencies obtained from parameterisation while ρ and X stand for density and the various components of water respectively.

4.1.1 Continuity equation

When the sources of mass are neglected, the continuity equation is expressed as,

$$\frac{\partial \rho}{\partial t} + div(\rho \mathbf{u}) = 0 \tag{4.1.1}$$

$$\frac{D\rho}{Dt} + \rho div \mathbf{u} = 0 \tag{4.1.2}$$

Equation (4.1.2) is the more fundamental form of continuity equation. $\mathbf{u}$ is the velocity of the rotating frame.

The spherical polar form of (4.1.2) is

$$\frac{D\rho}{Dt} + \rho\left(\frac{1}{r\cos\phi}\left[\frac{\partial u}{\partial \lambda} + \frac{\partial}{\partial \phi}(v\cos\phi)\right] + \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2 w\right)\right) = 0 \tag{4.1.3}$$

As starting point for transformation to a terrain-following coordinate system, the following alternative form is preferable.

$$\frac{D}{Dt}\left(\rho r^2 \cos\phi\right) + \rho r^2 \cos\phi\left(\frac{\partial}{\partial \lambda}\left[\frac{u}{r\cos\phi}\right] + \frac{\partial}{\partial \phi}\left[\frac{v}{r}\right] + \frac{\partial w}{\partial r}\right) = 0 \tag{4.1.4}$$

This can be written as

$$\frac{D}{Dt}\left(\rho r^2 \cos\phi\right) + \rho r^2 \cos\phi\left(\frac{\partial \dot{\lambda}}{\partial \lambda} + \frac{\partial \dot{\phi}}{\partial \phi} + \frac{\partial \dot{r}}{\partial r}\right) = 0 \tag{4.1.5}$$

Where $u = rcos\emptyset \frac{D\lambda}{Dt} = \dot{\lambda} rcos\emptyset,\ v = r\frac{D\emptyset}{Dt} = r\dot{\emptyset}\ and\ w = \frac{Dr}{Dt} = \dot{r}$

4.1.2 Equation of state

Given the pressure, p, density, ρ, and temperature, T, of a gas, the perfect gas law equation is given by

$$p = \rho RT \tag{4.1.6}$$

R is the gas constant for a unit mass of dry air.

It is convenient to work with the Exner function Π defined by

$$\Pi = \left(\frac{p}{p_0} \right)^{\frac{R}{C_p}} \tag{4.1.7}$$

Temperature T and potential temperature θ are related in the following way

$$\theta = \frac{T}{\Pi} \tag{4.1.8}$$

The pressure gradient terms in the component of momentum equation may be written in terms of θ rather than p for it varies far more rapidly with height.

$$\frac{1}{\rho}\frac{\partial p}{\partial X} = \frac{RT}{p}\frac{\partial p}{\partial X} = \frac{R\theta\Pi}{p}\frac{\partial p}{\partial X} = C_p\theta\frac{\partial \Pi}{\partial X} \tag{4.1.9}$$

Where $X = \lambda, \emptyset$ or r

(4.1.6) can be written in terms of Π and $k = \frac{R}{C_p}$ as follows

$$\Pi^{\frac{k-1}{k}}\rho\theta = \frac{p_0}{kC_p} \tag{4.1.10}$$

4.1.3 The first law of thermodynamics

It relates the change δU in internal energy of a mass of a fluid to the heating δQ and the work δW done by the mass of the fluid as follows.

$$\delta U = \delta Q - \delta W \tag{4.1.11}$$

Considering the pressure p of the mass of the fluid and the change in volume δV because of pressure p, then $\delta W = p\delta V$ and (4.1.11) becomes

$$\delta U + p\delta V = \delta Q \tag{4.1.12}$$

It is prudent to write (4.1.12) in terms of quantities per unit mass as follows

$$C_V \delta T + p\delta\alpha = \delta Q \tag{4.1.13}$$

C_V is specific heat at constant volume and $\alpha(=\frac{1}{\rho})$ is the specific volume. Hence

$$C_V \frac{DT}{Dt} + p\frac{D\alpha}{Dt} = \dot{Q} \tag{4.1.14}$$

$\dot{Q}$ is the rate of heating per unit mass, to which the element of the fluid is subject.

Considering the perfect gas, we have $p\alpha = RT$ and $C_p - C_V = R$, where C_p is the specific heat at constant pressure; (4.1.14) becomes

$$C_p \frac{DT}{Dt} - \alpha\frac{Dp}{Dt} = \dot{Q} \tag{4.1.15}$$

(4.1.15) can be written in terms of potential temperature defined by

$$\theta = T\left(\frac{p_0}{p}\right)^{\frac{R}{C_p}} \tag{4.1.16}$$

[Where p_0 is a reference pressure; conveniently, $p_0 = 1000hPa$]

(4.1.15) simplifies to

$$\frac{D\theta}{Dt} = \left(\frac{\theta}{T}\right)\frac{\dot{Q}}{C_p} \tag{4.1.17}$$

4.1.4. Navier-stokes equations

They generally have the following representation

$$\frac{\widehat{D}}{Dt}\hat{\mathbf{u}} = -\frac{1}{\rho}\mathbf{grad}p + \mathbf{G} \tag{4.1.18}$$

Where $\hat{\boldsymbol{u}} = \hat{\boldsymbol{u}}(r,t)$ is the velocity measured relative to an inertial frame;

$\rho = \rho(\boldsymbol{r},t)$ is density;

$p = p(\boldsymbol{r},t)$ is pressure;

$$\frac{\widehat{D}}{Dt} = \frac{\partial}{\partial t} + \left(\hat{\mathbf{u}}.\mathbf{grad}\right) \tag{4.1.19}$$

Equation (4.1.19) is the material rate of change as seen by an observer in an inertial frame. Finally, ***G*** includes all forces (per unit mass) except the pressure gradient force.

$\frac{1}{\rho}\mathbf{grad}p$ is the pressure gradient force per unit mass.

By using the relation between the rates of change of vectors seen in inertial and rotating frames, equation (4.1.18) can be converted to a form dealing with velocities $\mathbf{u}=\mathbf{u}(r,t)$ measured or defined relative to the rotating Earth.

$$\frac{\widehat{D}}{Dt}\mathbf{a}=\frac{D\mathbf{a}}{Dt}+\boldsymbol{\Omega}\times\mathbf{a} \tag{4.1.20}$$

Where $\frac{D}{Dt}$ is the material rate of change seen by an observer in a frame rotating relative to a "fixed star" with angular velocity, $\boldsymbol{\Omega}$.

Since $\hat{\mathbf{u}}=\frac{\widehat{D}}{Dt}\mathbf{r}$ and $\mathbf{u}\equiv\frac{D\mathbf{r}}{Dt}$ with $\mathbf{a}=\mathbf{r}=$ position vector relative to a point on the axis of rotation, (4.1.20) gives

$$\hat{\mathbf{u}}=\mathbf{u}+\boldsymbol{\Omega}\times\mathbf{r} \tag{4.1.21}$$

Applying (4.1.20) with $\mathbf{a}=\hat{\mathbf{u}}$ and using (4.1.21) gives

$$\frac{\widehat{D}\hat{\mathbf{u}}}{Dt}=\frac{D\mathbf{a}}{Dt}+2\boldsymbol{\Omega}\times\mathbf{u}+\boldsymbol{\Omega}\times(\boldsymbol{\Omega}\times\mathbf{r})+\dot{\boldsymbol{\Omega}}\times\mathbf{r} \tag{4.1.22}$$

Where $\dot{\boldsymbol{\Omega}}\equiv\frac{D\boldsymbol{\Omega}}{Dt}$.

$\dot{\boldsymbol{\Omega}}\times\mathbf{r}$ is negligible since astronomically detectable changes in magnitude and direction of the Earth's rotation vector are sufficiently small and slow. This renders (4.1.18) to be written as follows

$$\frac{D\mathbf{u}}{Dt}=-2\boldsymbol{\Omega}\times\mathbf{u}-\boldsymbol{\Omega}\times(\boldsymbol{\Omega}\times\mathbf{r})-\frac{1}{\rho}\mathbf{grad}p+\mathbf{G} \tag{4.1.23}$$

Where $-2\boldsymbol{\Omega}\times\mathbf{u}$ is the Coriolis force per unit mass and $-\boldsymbol{\Omega}\times(\boldsymbol{\Omega}\times\mathbf{r})$ is the Centrifugal force per unit mass.

In this case, the force per unit mass $\mathbf{G}$ includes the contributions of gravity, friction and electromagnetic forces. Only gravity and friction will be represented because the model is in the troposphere where electromagnetic effects are negligible.

Therefore,

$$\mathbf{G}=-\mathbf{grad}\Phi+\mathbf{S}^{\mathbf{u}}, \tag{4.1.24}$$

Where $\mathbf{S^u}$ is frictional force per unit mass and Φ is the true gravitational potential.

(4.1.23) becomes

$$\frac{D\mathbf{u}}{Dt} = -2\boldsymbol{\Omega}\times\mathbf{u} - \frac{1}{\rho}\mathbf{grad}p - \mathbf{grad}\Phi - \boldsymbol{\Omega}\times\left(\boldsymbol{\Omega}\times\mathbf{r}\right) + \mathbf{S^u} \tag{4.1.25}$$

But

$$\boldsymbol{\Omega}\times\left(\boldsymbol{\Omega}\times\mathbf{r}\right) = \boldsymbol{\Omega}\times\left(\boldsymbol{\Omega}\times\mathbf{s}\right) = -\Omega^2\mathbf{s} = -\mathbf{grad}\left(\frac{\Omega^2 s^2}{2}\right) \tag{4.1.26}$$

In terms of

$$\Phi_a = \Phi + \frac{1}{2}\Omega^2\mathbf{s}^2 \tag{4.1.27}$$

(4.1.25) becomes

$$\frac{D\mathbf{u}}{Dt} = -2\boldsymbol{\Omega}\times\mathbf{u} - \frac{1}{\rho}\mathbf{grad}p - \mathbf{grad}\Phi_a + \mathbf{S^u} \tag{4.1.28}$$

Where $\mathbf{grad}\Phi_a = g\mathbf{k}$ and Φ_a is the apparent gravitational potential.

We now decompose (4.1.28) into its spherical polar $(\lambda,\emptyset,r)$ components while recognising that the spherical polar system approximates the oblate spheroidal geopotential system.

It is noteworthy that

$$\lambda = longitude, \emptyset = latitude\ and$$

$$r = a + z \tag{4.1.29}$$

Where a is the Earth's mean radius and z is the distance above the sea mean level.

$$\frac{Du}{Dt} = -\frac{uw}{r} - 2\Omega w\cos\phi + \frac{uv\tan\phi}{r} + 2\Omega v\sin\phi - \frac{1}{\rho r\cos\phi}\frac{\partial p}{\partial\lambda} + S^u \tag{4.1.30}$$

$$\frac{Dv}{Dt} = -\frac{uw}{r} - \frac{u^2\tan\phi}{r} - 2\Omega u\sin\phi - \frac{1}{\rho}\frac{\partial p}{\partial\phi} + S^v \tag{4.1.31}$$

$$\frac{Dw}{Dt} = \frac{u^2+v^2}{r} + 2\Omega u\cos\phi - g - \frac{1}{\rho}\frac{\partial p}{\partial r} + S^w \tag{4.1.32}$$

Where

$$\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + \frac{u}{r\cos\phi}\frac{\partial}{\partial \lambda} + \frac{v}{r}\frac{\partial}{\partial \phi} + w\frac{\partial}{\partial r} \tag{4.1.33}$$

is the material derivative in spherical polar coordinates.

NB: The expressions of continuity equation, equation of state, first law of thermodynamics and the momentum equations above are for dry air. Representation of moisture will now follow.

4.1.5 Moisture representation

Moisture is mainly in 3 forms: water vapour, cloud liquid water and cloud frozen water. Precipitation is not explicitly treated. The following equation expresses the contributions of moisture.

$$\frac{Dm_X}{Dt} = S^{m_X} \tag{4.1.34}$$

m_X is the amount of water substance of type X associated with unit mass of dry air, $\frac{D}{Dt}$ is the material derivative and S^{m_X} represents the sources of water substance of type X.

If the mass of water substance of type X per unit volume of moist air is ρ_X, then

$$m_X \equiv \rho_X \Big/ \rho_y \tag{4.1.35}$$

Where ρ_y is the mass of dry air per unit volume of moist air. So m_X is the mixing ratio of water substance of type X with respect to dry air.

$$m_v \equiv \rho_v \Big/ \rho_y \tag{4.1.36}$$

$$m_{cl} \equiv \rho_{cl} \Big/ \rho_y \tag{4.1.37}$$

$$m_{cf} \equiv \rho_{cf} \Big/ \rho_y \tag{4.1.38}$$

Where (4.1.36), (4.1.37), (4.1.38) are the mixing ratios of water vapour, cloud liquid water and cloud frozen water respectively.

The masses per unit volume are related in the following way

$$\rho = \rho_y + \rho_v + \rho_{cl} + \rho_{cf} \tag{4.1.39}$$

Therefore, the budget equations for various phases of water are given by

$$\frac{Dm_X}{Dt} = S^{m_X} \tag{4.1.40}$$

where $X = v,\ cl$ or cf.

According to *Dalton's Law of Partial Pressures,* the pressure exerted by a mixture of dry air and water vapour is equal to the sum of the pressures separately. If R_d and R_v are the gas constants (per unit mass) for dry air and water vapour, and $\theta \equiv R_d / R_v$, we find

$$p = p_y + p_v = \left(\rho_y R_d + \rho_v R_v\right)T = \rho R_d T\left(\frac{\rho_y}{\rho} + \frac{\rho_v R_v}{\rho R_d}\right) \tag{4.1.41}$$

Or

$$p = \rho R_d T_v \tag{4.1.42}$$

Where

$$T_v = T\left(\frac{1 + \frac{1}{\varepsilon} m_v}{1 + m_v + m_{cl} + m_{cf}}\right) \tag{4.1.43}$$

T_v is the virtual temperature i.e. the temperature that air would have to have, at a given density in order to exert the same pressure as the mixture of dry air and water substance at temperature T.

Taking into consideration moisture representation, the primitive equations take the following form:

(i) Navier-Stokes Equations

$$\frac{Du}{Dt} = -\frac{uw}{r} - 2\Omega w\cos\phi + \frac{uv\tan\phi}{r} + 2\Omega v\sin\phi - \frac{C_{pd}\theta_V}{r\cos\phi}\frac{\partial \Pi}{\partial \lambda} + S^u \tag{4.1.44}$$

$$\frac{Dv}{Dt} = -\frac{vw}{r} - \frac{u^2\tan\phi}{r} - 2\Omega u\sin\phi - \frac{C_{pd}\theta_V}{r}\frac{\partial \Pi}{\partial \phi} + S^v \tag{4.1.45}$$

$$\frac{Dw}{Dt} = \frac{(u^2 + v^2)}{r} - 2\Omega u\cos\phi - g - C_{pd}\theta_V\frac{\partial \Pi}{\partial r} + S^w \tag{4.1.46}$$

Where

$$\Pi = \left(\frac{p}{p_0}\right)^{\frac{R_d}{C_{pd}}} \tag{4.1.47}$$

$$\theta_V = \frac{T}{\prod}\left(\frac{1+\frac{1}{\varepsilon}m_v}{1+m_v+m_{cl}+m_{cf}}\right) \tag{4.1.48}$$

(ii) Continuity Equation

$$\frac{D}{Dt}\left(\rho_y r^2\cos\phi\right)+\rho_y r^2\cos\phi\left(\frac{\partial}{\partial\lambda}\left[\frac{u}{r\cos\phi}\right]+\frac{\partial}{\partial\phi}\left[\frac{v}{r}\right]+\frac{\partial w}{\partial r}\right)=0 \tag{4.1.49}$$

Where

$$\rho = \rho_y\left(1+m_v+m_{cl}+m_{cf}\right) \tag{4.1.50}$$

(iii) First Law of Thermodynamics

$$\frac{D\theta}{Dt} = S^{\theta} = \left(\frac{\theta}{T}\right)\frac{\dot{Q}}{C_{pd}} \tag{4.1.51}$$

Where

$$\theta = \frac{T}{\prod} = T\left(\frac{p_0}{p}\right)^{\frac{R_d}{C_{pd}}} \tag{4.1.52}$$

(iv) Equation of State

$$\prod^{\frac{k_d-1}{k_d}}\rho\theta_v = \frac{p_0}{k_d C_{pd}}, \left[k_d \equiv \frac{R_d}{C_{pd}}\right] \tag{4.1.53}$$

(v) Representation of Moisture

$$\frac{Dm_X}{Dt} = S^{m_X} \tag{4.1.54}$$

where $X = v,\ cl$ or cf .

4.2. Momentum equations in a rotated system

The Coriolis terms in the momentum equations are changed when written in terms of rotated longitude and latitude.

Let the geographical latitude of the rotated pole be ϕ_0. Then the Earth's rotation vector has latitude ϕ_0 and longitude zero in the rotated system. If **I**, **J**, **K** be the unit vectors in the directions Ox, Oy, Oz in the rotated systems, then

$$\Omega = \mathbf{I}\Omega\cos\phi_0 + \mathbf{k}\Omega\sin\phi_0 \qquad (4.2.1)$$

And

$$\mathbf{u} = u\mathbf{i} + v\mathbf{j} + w\mathbf{k} \qquad (4.2.2)$$

i, **j** and **k** are unit vectors in the zonal (λ), meridional (ϕ) and radial (r) directions in the rotated system as shown in figure (4.2.1).

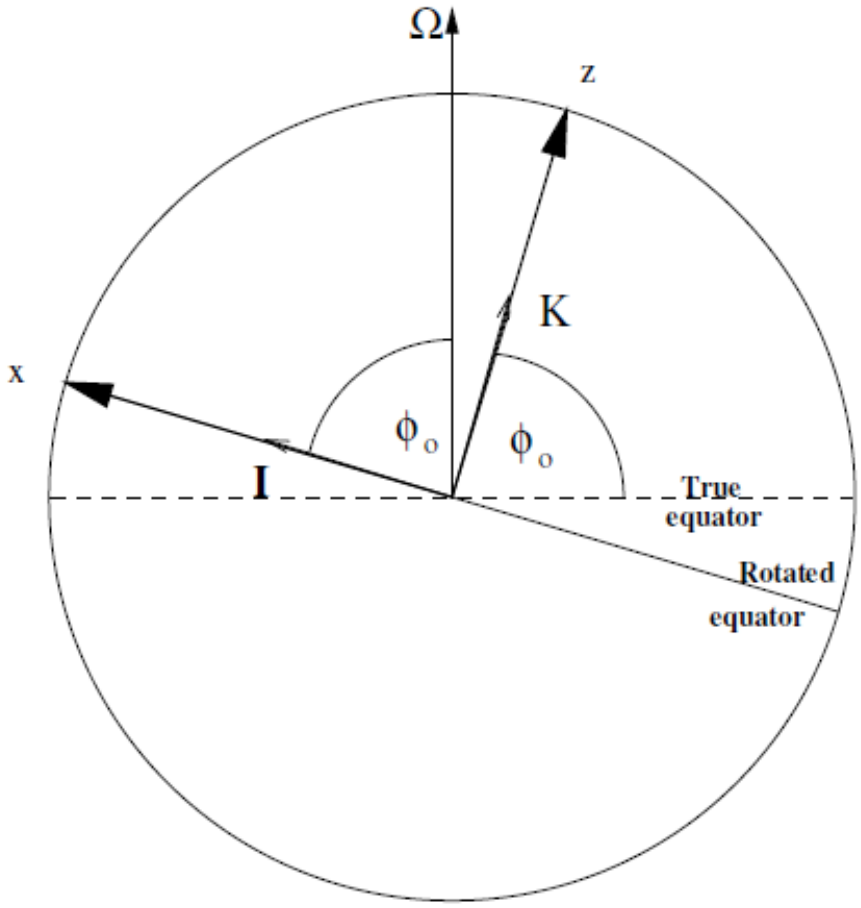


Figure 2: Rotated section of a sphere about the Earth's rotation vector

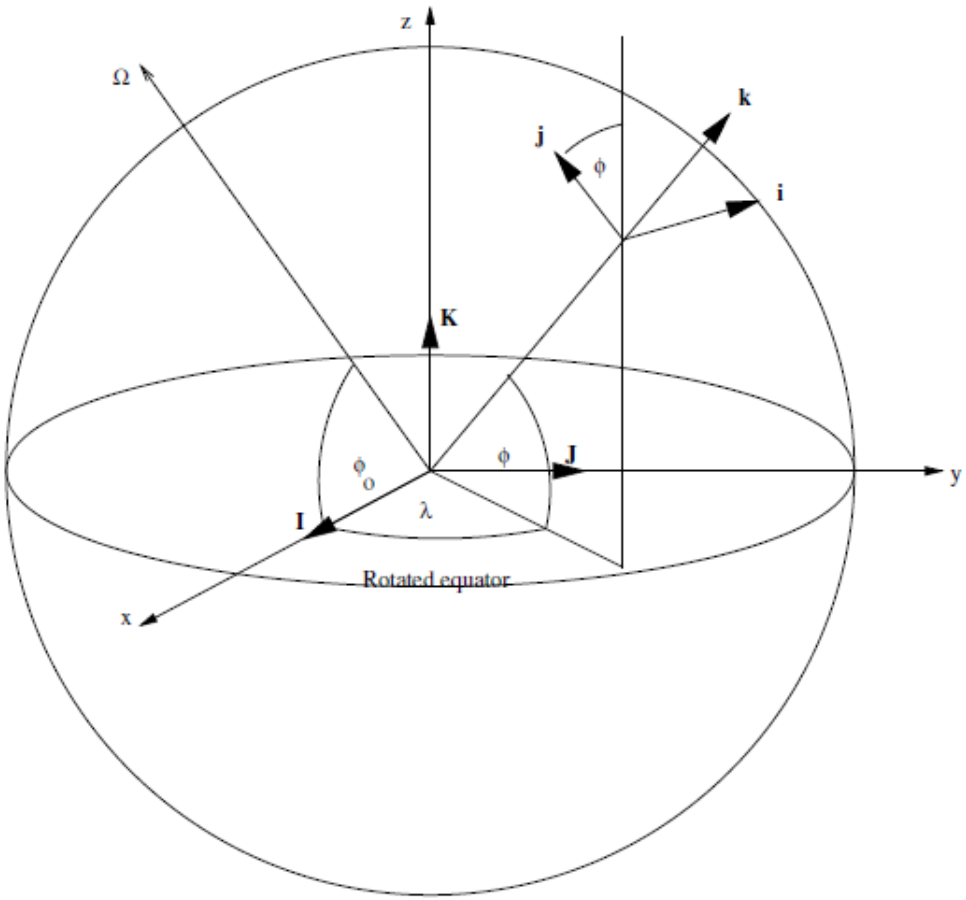


Figure 3: The Unit vectors **I, J, k** in the directions *Ox, Oy, Oz* in the rotated system, and the unit vectors **i, j, k** associated with the zonal, meridional and radial directions.

From figure (2) we can express **i, j** and **k** in terms of **I, J,** and **K:**

$$\mathbf{i} = -\mathbf{I}\sin\lambda + \mathbf{J}\cos\lambda \tag{4.2.3}$$

$$\mathbf{j} = -\mathbf{I}\sin\phi\cos\lambda - \mathbf{J}\sin\phi\sin\lambda + \mathbf{K}\cos\phi \tag{4.2.4}$$

$$\mathbf{k} = \mathbf{I}\cos\phi\cos\lambda + \mathbf{J}\cos\phi\sin\lambda + \mathbf{K}\sin\phi \tag{4.2.5}$$

$$\boldsymbol{\Omega} = \Omega_\lambda \mathbf{i} + \Omega_\phi \mathbf{j} + \Omega_r \mathbf{k} \tag{4.2.6}$$

From equation (4.2.1) – (4.2.5), we get:

$$\Omega_\lambda = \boldsymbol{\Omega}.\mathbf{i} = -\Omega\sin\lambda\cos\phi_0 \equiv \frac{1}{2} f_1 \tag{4.2.7}$$

$$\Omega_\phi = \boldsymbol{\Omega}.\mathbf{j} = \Omega\left(\cos\phi\sin\phi_0 - \sin\phi\cos\lambda\cos\phi_0\right) \equiv \frac{1}{2} f_2 \tag{4.2.8}$$

$$\Omega_z = \boldsymbol{\Omega}.\mathbf{k} = \Omega\left(\sin\phi\sin\phi_0 + \cos\phi\cos\lambda\cos\phi_0\right) \equiv \frac{1}{2} f_3 \tag{4.2.9}$$

Therefore, the Coriolis term in the rotated system can be represented as follows:

$$\begin{aligned} -2\boldsymbol{\Omega}\times\mathbf{u} &= \left(u\mathbf{i}+v\mathbf{j}+w\mathbf{k}\right)\times\left(f_1\mathbf{i}+f_2\mathbf{j}+f_3\mathbf{k}\right) \\ &= \left(f_3v-f_2w\right)\mathbf{i}+\left(f_1w-f_3u\right)\mathbf{j}+\left(f_2u-f_1v\right)\mathbf{k} \end{aligned} \tag{4.2.10}$$

With these considerations, the momentum equations in the rotated system will take the following forms:

$$\frac{Du}{Dt} = -\frac{uw}{r}+\frac{uv\tan\phi}{r}+f_3v-f_2w-\frac{C_{pd}\theta_v}{r\cos\phi}\frac{\partial\Pi}{\partial\lambda}+S^u \tag{4.2.11}$$

$$\frac{Dv}{Dt} = -\frac{uw}{r}-\frac{u^2\tan\phi}{r}+f_1w-f_3u-\frac{C_{pd}\theta_v}{r}\frac{\partial\Pi}{\partial\phi}+S^v \tag{4.2.12}$$

$$\frac{Dw}{Dt} = \frac{\left(u^2+v^2\right)}{r}+f_2u-f_1v-g\left(1+q_{cl}+q_{cf}\right)-C_{pd}\theta_v\frac{\partial\Pi}{\partial r}+S^w \tag{4.2.13}$$

In this case

$$f_1 = -2\Omega\sin\lambda\cos\phi_0$$

$$f_2 = 2\Omega\left(\cos\phi\sin\phi_0-\sin\phi\cos\lambda\cos\phi_0\right)$$

$$f_3 = 2\Omega\left(\sin\phi\sin\phi_0+\cos\phi\cos\lambda\cos\phi_0\right)$$

4.3. Discretisation of Governing Equations

To discretise governing equations we are going to use Semi-Implicit Semi-Lagrangian time discretisation. The first part of this work is devoted to highlighting how this method works. The strengths of this method over the Eulerian decomposition are also given.

4.3.1 Off-centred, Semi-implicit, Semi-Lagrangian(SISL) time discretisation

Unlike in the Eulerian decomposition where the material derivatives are separated into local rates of change and advection terms, we use the Semi-Lagrangian treatment where the material derivatives are retained intact, and next time-step values at grid points are found by intergrating along interpolated trajectories.

4.3.1.1 Advantages of Semi-Lagrangian technique

1. Stability even when long time steps are taken.

2. It doesn't have Eulerian advection terms.

It is used for the momentum, thermodynamic and moisture equations.

It may be subject to numerical instabilities if certain extrapolation procedures are used.

4.3.2 Outline of the method

Consider the following equation

$$\frac{DF}{Dt} = \Psi \tag{4.3.1}$$

In this case, $\frac{D}{Dt}$ is the material derivative, F is a scalar variable and Ψ is a source term.

We integrate (4.3.1) between times $t^n = n\Delta t$ and $t^{n+1} = t^n + \Delta t$ following a parcel of air that arrives at grid point x_a at time t^{n+1}. The change in t for the parcel that arrives at x_a at time t^{n+1} is simply the integral of Ψ along its trajectory over the relevant time interval:

$$F^{n+1} - F_d^n = \int_{t^n}^{t^n+\Delta t} \Psi dt = \overline{\Psi}\Delta t \tag{4.3.2}$$

F^{n+1} is the value of F at time t^{n+1} at the arrival grid point x_a, i.e.

$$F^{n+1} \equiv F(x_a, t^{n+1}) \tag{4.3.3}$$

And F_d^n is the value of F for the same parcel of air but at time t^n, i.e.

$$F_d^n \equiv F(x_a, t^n) \tag{4.3.4}$$

Where x_d is the location of the parcel at time t^n (the departure point of the parcel).

$\overline{\Psi}$ is the time average of Ψ along the trajectory from the departure point x_d to the arrival point x_a.

x_d and F_d^n and $\overline{\Psi}$ have to be estimated from the available gridpoint values.

Equation (4.3.2) contains no Eulerian advection terms, is an exact integral of (4.3.1) and it involves no truncation errors. However, errors are introduced through the estimation of x_d, F_d^n and trajectory time-average $\overline{\Psi}$. These estimates require integration and interpolation.

(4.3.2) represents a two-time-level scheme, Δt being the time step. Two-time-level schemes require less storage, and for a given time step they reach a given forecast time in 50% fewer steps because successive intervals do not overlap.

The trajectory time-average $\overline{\Psi}$ may be approximated by a weighted average of the values of Ψ at the departure and arrival points:

$$\overline{\Psi} \equiv \alpha\Psi^{n+1} + (1-\alpha)\Psi_d^n \tag{4.3.5}$$

α is the trajectory weighting factor. If Ψ involves F, $\alpha \geq \frac{1}{2}$ is a necessary condition for stability.

For an off-centred two-time level scheme, $\frac{1}{2} < \alpha \leq 1$, the truncation error becomes $o(\Delta t)$ and (4.3.2) becomes

$$F^{n+1} - F_d^n = \Delta t\left[\alpha\Psi^{n+1} + (1-\alpha)\Psi_d^n\right] \tag{4.3.6}$$

This off-centred two-time-level scheme is generally more accurate and less damping the closer α is to $\frac{1}{2}$.

By grouping terms at the new time t^{n+1} on the left side and known quantities on the right, (4.3.6) may be written as

$$F^{n+1} - \alpha\Delta t\Psi = F_d^n + (1-\alpha)\Delta t\Psi_d^n \equiv \left[F + (1-\alpha)\Psi\right]_d^n \tag{4.3.7}$$

The term $-\alpha\Delta t\Psi^{n+1}$ involves the forcing evaluated at the arrival point at the new time-level. The presence of this term complicates the calculation of F^{n+1} especially if all or part of Ψ is non-linear in F or in any of the prognostic variables of the model. The part of Ψ that is linear in F or in any of the prognostic variables can be dealt with by algebraic elimination. The part of Ψ^{n+1} that are non-linear in F^{n+1} have to be accommodated using some iterative procedure, which in practice consist of a fixed number of "predictor-corrector" steps. (4.3.7) is referred to as an off-centred, semi-implicit, semi-Lagrangian form.

Evaluation of the departure point quantities F_d^n and Ψ_d^n involves approximation in two stages:

1. Location of departure point x_d; and

2. Interpolation to obtain $F_d^n \equiv F(x_d, t^n)$ and $\Psi_d^n \equiv \Psi^n\left(x_d, t^n\right)$ from available grid point values of Ψ and F at time-level n.

The departure-point calculation exploits the definition of the continuously-varying velocity field $\mathbf{u}$ as the rate of change of the position x of parcels of air relative to the rotating Earth.

$$\frac{D\mathbf{x}(t)}{Dt} = \mathbf{u}\left(x(t), t\right) \tag{4.3.8}$$

In the integrand form, this becomes

$$\mathbf{x}_a - \mathbf{x}_d = \int_{t^n}^{t^n+\Delta t} \mathbf{u}dt = \bar{\mathbf{u}}\Delta t \tag{4.3.9}$$

Where the integrand $\mathbf{u}$ is evaluated along the trajectory between the departure point x_d and the arrival point, x_a.

4.3.3 Semi-Lagrangian treatment of the momentum equation in spherical geometry

Momentum equation is expressed as

$$\frac{D\mathbf{u}}{Dt} = \mathbf{\Psi} \tag{4.3.10}$$

Where $\mathbf{\Psi}$ represent the coriolis, centrifugal, pressure gradient and frictional forces. The result of integrating (4.3.10) along trajectories the same way as (4.3.2) is given by

$$\mathbf{u}^{n+1} - \mathbf{u}_d^n = \bar{\mathbf{\Psi}}\Delta t \tag{4.3.11}$$

Where

$$\mathbf{\Psi} = -2\mathbf{\Omega}\times\mathbf{u} - g\mathbf{k} - \frac{1}{\rho}\mathbf{grad}p + \mathbf{S}^{\mathbf{u}} \tag{4.3.12}$$

We now need to isolate (4.3.10) to its zonal meridional and radial components at the arrival points $\mathbf{x}_a$.

The unit vectors $\mathbf{i}_a$, $\mathbf{j}_a$, $\mathbf{k}_a$ in the zonal, meridional and radial directions at the arrival point (λ_a, ϕ_a, r_a) may be expressed in terms of the unit vectors $\mathbf{I}$, $\mathbf{J}$, $\mathbf{K}$ in a geocentric Cartesian system as

$$\mathbf{i}_a = -\mathbf{I}\sin\lambda_a + \mathbf{J}\cos\lambda_a \tag{4.3.13}$$

$$\mathbf{j}_a = -\mathbf{I}\sin\phi_a\cos\lambda a - \mathbf{J}\sin\phi_a\sin\lambda_a + \mathbf{K}\cos\phi_a \tag{4.3.14}$$

$$\mathbf{k}_a = \mathbf{I}\cos\phi_a\cos\lambda_a + \mathbf{J}\cos\phi_a\sin\lambda_a + \mathbf{K}\sin\phi_a \tag{4.3.15}$$

Similar expressions can be obtained for departure point.

The departure and arrival point velocities are as follows

$$\mathbf{u}_d^n = u_d^n i_a + v_d^n j_d + w_d^n k_d \tag{4.3.16}$$

$$\mathbf{u}^{n+1} = u^{n+1} i_a + v^{n+1} j_a + w^{n+1} k_a \tag{4.3.17}$$

The following expressions are then obtained from (4.3.10) through scalar multiplication by the arrival point unit vectors.

$$u^{n+1} = \mathbf{i}_a.\mathbf{u}^{n+1} = \mathbf{i}_a.\mathbf{u}_d^n + \mathbf{i}_a.\overline{\mathbf{\Psi}}\Delta t \tag{4.3.18}$$

$$v^{n+1} = \mathbf{j}_a.\mathbf{u}^{n+1} = \mathbf{j}_a.\mathbf{u}_d^n + \mathbf{j}_a.\overline{\mathbf{\Psi}}\Delta t \tag{4.3.19}$$

$$w^{n+1} = \mathbf{k}_a.\mathbf{u}^{n+1} = \mathbf{k}_a.\mathbf{u}_d^n + \mathbf{k}_a.\overrightarrow{\mathbf{\Psi}}\Delta t \tag{4.3.20}$$

Using (4.3.13)-(4.3.20) we write the components of velocity in the arrival point in terms of components of velocity at the departure point.

$$\begin{aligned}&\mathbf{i}_a.\mathbf{u}_d^n = \mathbf{i}_a.(u_d^n\mathbf{i}_d + v_d^n\mathbf{j}_d + w_d^n\mathbf{j}_d)\\&= u_d^n\cos(\lambda_a - \lambda_d) + v_d^n\sin\phi_d\sin(\lambda_a - \lambda_d) - w_d^n\cos\phi_d\sin(\lambda_a - \lambda_d)\end{aligned} \tag{4.3.21}$$

(4.3.21) can be written as

$$u^{n+1} - m_{uu}u_d^n = m_{uv}v_d^n + m_{uw}w_d^n + \mathbf{i}_a.\overline{\mathbf{\Psi}}\Delta t \tag{4.3.22}$$

where

$$m_{uu} = \cos(\lambda_a - \lambda_d), m_{uv} = \sin\phi_d\sin(\lambda_a - \lambda_d), m_{uw} = -\cos\phi_d\sin(\lambda_a - \lambda_d) \tag{4.3.23}$$

(4.3.19) and (4.3.20) can similarly be written as follows

$$v^{n+1} - m_{vv}v_d^n = m_{vu}u_d^n + m_{vw}w_d^n + \mathbf{j}_a.\overline{\mathbf{\Psi}}\Delta t \tag{4.3.24}$$

$$w^{n+1} - m_{ww}w_d^n = m_{wu}u_d^n + m_{wv}u_d^n + \mathbf{k}_a.\overline{\mathbf{\Psi}}\Delta t \tag{4.3.25}$$

Here

$$m_{vu} = -\sin\phi_a\sin(\lambda_a - \lambda_d) \tag{4.3.26}$$

$$m_{vv} = \cos\phi_a cos\phi_d + \sin\phi_a\sin\phi_d\cos(\lambda_a - \lambda_d) \tag{4.3.27}$$

$$m_{vw} = \cos\phi_a\sin\phi_d - \sin\phi_a\cos\phi_d\cos(\lambda_a - \lambda_d) \tag{4.3.28}$$

$$m_{wu} = \cos\phi_a\sin(\lambda_a - \lambda_d) \tag{4.3.29}$$

$$m_{wv} = \sin\phi_a\cos\phi_d - \cos\phi_a\sin\phi_d\cos(\lambda_a - \lambda_d) \tag{4.3.30}$$

$$m_{ww} = \sin\phi_a\sin\phi_d + \cos\phi_a\cos\phi_d\cos(\lambda_a - \lambda_d) \tag{4.3.31}$$

The matrix **M** is a finite rotation matrix. It is also orthogonal for its inverse is its transpose. The off-diagonal elements correspond to the metric terms of the momentum equations. $m_{uv}, m_{uw}, m_{vu}, m_{vw}, m_{wu}$ and m_{wv} correspond respectively to

$\frac{(uv\tan\phi)}{r}, \frac{-wu}{r}, -\frac{u^2\tan\phi}{r}, -\frac{vw}{r}, \frac{u^2}{r}$ and $\frac{v^2}{r}$. These correspondences can be established by considering the limit $\Delta t \to 0$; then $\lambda_d \to \lambda_a$ and $\phi_d \to \phi_a$.

It now remains to discretize the source term. We begin from the following time-average.

$$\overline{\boldsymbol{\Psi}} \equiv \alpha \boldsymbol{\Psi}^{n+1} + (1-\alpha)\boldsymbol{\Psi}_d^n \tag{4.3.32}$$

The source term is expressed in terms of unit vectors at the arrival and departure points as follows:

$$\overline{\boldsymbol{\Psi}} \equiv \alpha\left(\Psi_\lambda^{n+1}\mathbf{i}_a + \Psi_\phi^{n+1}\mathbf{j}_a + \Psi_r^{n+1}\mathbf{k}_a\right) + (1-\alpha)\left(\Psi_{d\lambda}^n\mathbf{i}_d + \Psi_{d\phi}^n\mathbf{j}_d + \Psi_{dr}^n\mathbf{k}_d\right) \tag{4.3.33}$$

Therefore

$$\mathbf{i}_a.\overline{\boldsymbol{\Psi}} \equiv \alpha\Psi_d^{n+1} + (1-\alpha)\left(\Psi_{d\lambda}^n\mathbf{i}_a.\mathbf{i}_d + \Psi_{d\phi}^n\mathbf{i}_a.\mathbf{j}_d + \Psi_{dr}^n\mathbf{i}_a.\mathbf{k}_d\right) \tag{4.3.34}$$

$$\mathbf{j}_a.\overline{\boldsymbol{\Psi}} \equiv \alpha\Psi_\phi^{n+1} + (1-\alpha)\left(\Psi_{d\lambda}^n\mathbf{j}_a.\mathbf{i}_d + \Psi_{d\phi}^n\mathbf{j}_a.\mathbf{j}_d + \Psi_{dr}^n\mathbf{j}_a.\mathbf{k}_d\right) \tag{4.3.35}$$

$$\mathbf{k}_a.\overline{\boldsymbol{\Psi}} \equiv \alpha\Psi_r^{n+1} + (1-\alpha)\left(\Psi_{d\lambda}^n\mathbf{k}_a.\mathbf{i}_d + \Psi_{d\phi}^n\mathbf{k}_\phi.\mathbf{j}_d + \Psi_{dr}^n\mathbf{k}_a.\mathbf{k}_d\right) \tag{4.3.36}$$

The scalar products in equations (4.3.34)-(4.3.36) are elements of the finite rotation matrix **M**.

Let $\beta = (1-\alpha)$. We can rewrite (4.3.34)-(4.3.36) a follows.

$$u^{n+1} - \alpha\Psi_\lambda^{n+1}\Delta t = m_{uu}\left(u_d^n + \beta\Psi_{d\lambda}^n\Delta t\right) + m_{uv}\left(v_d^n + \beta\Psi_{d\phi}^n\Delta t\right) + m_{uw}\left(w_d^n + \beta\Psi_{dr}^n\Delta t\right) \tag{4.3.37}$$

$$v^{n+1} - \alpha\Psi_\phi^{n+1}\Delta t = m_{vu}\left(u_d^n + \beta\Psi_{d\lambda}^n\Delta t\right) + m_{vu}\left(v_d^n + \beta\Psi_{d\phi}^n\Delta t\right) + m_{vw}\left(w_d^n + \beta\Psi_{dr}^n\Delta t\right) \tag{4.3.38}$$

$$w^{n+1} - \alpha\Psi_r^{n+1}\Delta t = m_{wu}\left(u_d^n + \beta\Psi_{d\lambda}^n\Delta t\right) + m_{wv}\left(v_d^n + \beta\Psi_{d\phi}^n\Delta t\right) + m_{ww}\left(w_d^n + \beta\Psi_{dr}^n\Delta t\right) \tag{4.3.39}$$

Or, most precisely, equations (4.3.37)-(4.3.39) can be written compactly in vector-matrix as follows:

$$\mathbf{u}^{n+1} - \alpha\boldsymbol{\Psi}^{n+1}\Delta t = \mathbf{M}\left(u_d^n + (1-\alpha)\boldsymbol{\Psi}_d^n\Delta t\right) \tag{4.3.40}$$

M transforms vectors between the departure and arrival point systems.

The source $\boldsymbol{\Psi}$ may be represented as sum of parts $\boldsymbol{\Psi}_\mathbf{k}$ with the associated weighting factor α_k. Thus (4.3.40) can be expressed as follows:

$$\mathbf{u}^{n+1} - \sum_k \alpha_k \boldsymbol{\Psi}_k^{n+1}\Delta t = \mathbf{M}\left\{\mathbf{u}_d^n + \sum_k (1-\alpha_k)\boldsymbol{\Psi}_{kd}^n\Delta t\right\} \tag{4.3.41}$$

This renders it easy to represent different terms in the momentum equation with different trajectory weighting factors α_k.

With these very vital expressions, we can now embark on discretisation of governing equations that were developed earlier.

4.3.4 Discretisation of the u-component of the momentum equation

The equation to be discretised is the following:

$$\frac{Du}{Dt} - f_3 v + f_2 w - \frac{uv\tan\phi}{r} + \frac{uw}{r} + \frac{C_{pd}\theta_v}{r\cos\theta}\left(\frac{\partial\Pi}{\partial\lambda} - \frac{\partial\Pi}{\partial r}\frac{\partial r}{\partial\lambda}\right) = S^u \tag{4.3.42}$$

(i) At levels $k = \frac{3}{2}, \frac{5}{2}, ..., N - \frac{3}{2}$

(4.3.42) is discretised using a 2-time-level, off-centred, semi-implicit, semi-Lagrangian scheme at point u($\lambda_1, \phi_{j-\frac{1}{2}}, \eta_{k-\frac{1}{2}}$) to yield:

$$\begin{aligned}
\frac{u^{n+1} - u_d^n}{\Delta t} &= \alpha_3\left[f_3\overline{v}^{\lambda\phi} - \frac{C_{pd}}{\overline{r}^{\lambda}\cos\phi}\left(\overline{\theta}_v^{r\lambda}\frac{\partial\Pi}{\partial\lambda} - \overline{\theta_v\frac{\partial\Pi}{\partial r}}^{r\lambda}\frac{\partial r}{\partial\lambda}\right)\right]^{n+1} \\
&+ (1-\alpha_3)\left[f_3\overline{v}^{\lambda\phi} - \frac{C_{pd}}{\overline{r}^{\lambda}\cos\phi}\overline{\theta}_v^{r\lambda}\frac{\partial\Pi}{\partial\lambda} - \overline{\theta_v\frac{\partial\Pi}{\partial r}}^{r\lambda}\frac{\partial r}{\partial\lambda}\right]_d^n \\
&- \alpha_4\left[f_2\overline{w}^{r\lambda}\right]^{n+1} - (1-\alpha_4)\left[f_2\overline{w}^{r\lambda}\right]_d^n \\
&+ \alpha_p\left[S^u\right]^{n+1} + (1-\alpha_p)\left[S^u\right]_d^n
\end{aligned} \tag{4.3.43}$$

In this expression, as in all others, the following expressions are noteworthy.

$$\left(\overline{F}^{\lambda}\right)_{i,j} = \left(\frac{\lambda_{i+\frac{1}{2}} - \lambda_i}{\Delta\lambda_i}\right)F_{i-\frac{1}{2},j} + \left(\frac{\lambda_i - \lambda_{i-\frac{1}{2}}}{\Delta\lambda_i}\right)F_{i+\frac{1}{2},j} \tag{4.3.44}$$

$$\left(\overline{F}^{\phi}\right)_{i,j} = \left(\frac{\phi_{j+\frac{1}{2}} - \phi_j}{\Delta\phi_j}\right)F_{i,j-\frac{1}{2}} + \left(\frac{\phi_j - \phi_{j-\frac{1}{2}}}{\Delta\phi_j}\right)F_{i,j+\frac{1}{2}} \tag{4.3.45}$$

$$
\begin{aligned}
\left(\overline{F}^{\lambda\phi}\right)_{i,j} &= \left[\overline{\left(\overline{F}^{\lambda}\right)^{\phi}}\right]_{i,j} \\
&= \left(\frac{\phi_j - \phi_{j-\frac{1}{2}}}{\Delta\phi_j}\right)\left[\left(\frac{\lambda_{i+\frac{1}{2}} - \lambda_i}{\Delta\lambda_i}\right)F_{i-\frac{1}{2},j+\frac{1}{2}} + \left(\frac{\lambda_i - \lambda_{i-\frac{1}{2}}}{\Delta\lambda_i}\right)F_{i+\frac{1}{2},j+\frac{1}{2}}\right] \\
&+ \left(\frac{\phi_{j+\frac{1}{2}} - \phi_j}{\Delta\phi_j}\right)\left[\left(\frac{\lambda_{i+\frac{1}{2}} - \lambda_i}{\Delta\lambda_i}\right)F_{i-\frac{1}{2},j-\frac{1}{2}} + \left(\frac{\lambda_i - \lambda_{i-\frac{1}{2}}}{\Delta\lambda_i}\right)F_{i+\frac{1}{2},j-\frac{1}{2}}\right] \\
&= \left(\overline{F}^{\phi\lambda}\right)_{i,j}
\end{aligned}
\tag{4.3.46}
$$

$$
\overline{F_k}^{r} = \frac{\left(r_{i,j,k} - r_{i,j,k-\frac{1}{2}}\right)F\left(r_{i,j,k+\frac{1}{2}}\right) + \left(r_{i,j,k+\frac{1}{2}} - r_{i,j,k}\right)F\left(r_{i,j,k-\frac{1}{2}}\right)}{r_{i,j,k+\frac{1}{2}} - r_{i,j,k-\frac{1}{2}}} \tag{4.3.47}
$$

$$
\overline{F_k}^{\eta} = \frac{\left(\eta_k - \eta_{k-\frac{1}{2}}\right)F\left(\eta_{k+\frac{1}{2}}\right) + \left(\eta_{k+\frac{1}{2}} - \eta_k\right)F\left(\eta_{k-\frac{1}{2}}\right)}{\eta_{k+\frac{1}{2}} - \eta_{k-\frac{1}{2}}} \tag{4.3.48}
$$

$$
\delta_\lambda F\left(\lambda_i,\phi_j\right) \equiv \left(\delta_\lambda F\right)_{i,j} = \frac{F\left(\lambda_{i+\frac{1}{2}},\phi_j\right) - F\left(\lambda_{i-\frac{1}{2}},\phi_j\right)}{\lambda_{i+\frac{1}{2}} - \lambda_{i-\frac{1}{2}}} \equiv \frac{F_{i+\frac{1}{2},j} - F_{i-\frac{1}{2},j}}{\Delta\lambda_i} \tag{4.3.49}
$$

$$
\delta_\phi F\left(\lambda_i,\phi_j\right) \equiv \left(\delta_\phi F\right)_{i,j} = \frac{F\left(\lambda_i,\phi_{j+\frac{1}{2}}\right) - F\left(\lambda_i,\phi_{j-\frac{1}{2}}\right)}{\phi_{j+\frac{1}{2}} - \phi_{j-\frac{1}{2}}} \equiv \frac{F_{i,j+\frac{1}{2}} - F_{i,j-\frac{1}{2}}}{\Delta\phi_j} \tag{4.3.50}
$$

$$
\delta_r F\left(r_{i,j,k}\right) \equiv \left(\delta_r F\right)_k = \frac{F\left(r_{i,j,k+\frac{1}{2}}\right) - F\left(r_{i,j,k-\frac{1}{2}}\right)}{r_{i,j,k+\frac{1}{2}} - r_{i,j,k-\frac{1}{2}}} \equiv \frac{F_{k+\frac{1}{2}} - F_{k-\frac{1}{2}}}{r_{i,j,k+\frac{1}{2}} - r_{i,j,k-\frac{1}{2}}} \tag{4.3.51}
$$

$$
\delta_{2r} F\left(r_{i,j,k}\right) \equiv \left(\delta_{2r} F\right)_k = \frac{F\left(r_{i,j,k+1}\right) - F\left(r_{i,j,k-1}\right)}{r_{i,j,k+1} - r_{i,j,k-1}} \equiv \frac{F_{k+1} - F_{k-1}}{r_{i,j,k+1} - r_{i,j,k-1}} \tag{4.3.52}
$$

Due to the complexity associated with the time-implicit treatment of $f_2 w$ Coriolis term, the non-linear pressure gradient terms and the forcing term, S^u, the predictor corrector method is developed below.

1.) Predictor step

Let $\tilde{u}^{(1)}$ be the first predictor for u^{n+1}. We first neglect the forcing term, S^u, and then replace all the remaining terms evaluated at mesh points at time $(n+1)\Delta t$ in (1.1.43) by their values at the same mesh points but at time $n\Delta t$.

Thus,

$$\begin{aligned}\frac{\tilde{u}^{(1)}-u_d^n}{\Delta t} &= \alpha_3\left[f_3\bar{v}^{-\lambda\phi} - \frac{C_{pd}}{\bar{r}^{\lambda}\cos\phi}\left(\bar{\theta}_v^{r\lambda}\delta_\lambda\Pi - \overline{\theta_v\delta_r\Pi}^{r\lambda}\delta_\lambda r\right)\right]^n \\ &+(1-\alpha_3)\left[f_3\bar{v}^{-\lambda\phi} - \frac{C_{pd}}{\bar{r}^{\lambda}\cos\phi}\left(\bar{\theta}_v^{r\lambda}\delta_\lambda\Pi - \overline{\theta_v\delta_r\Pi}^{r\lambda}\delta_\lambda r\right)\right]_d^n \\ &-\alpha_4\left[f_2\bar{w}^{r\lambda}\right]^n - (1-\alpha_4)\left[f_2\bar{w}^{r\lambda}\right]_d^n\end{aligned} \tag{4.3.53}$$

Equivalently, let $R_u \equiv \tilde{u}^{(1)} - u^n$. This step is actually the computation of R_u and is given by:

$$\begin{aligned}R_u = &-\left\{u-\alpha_3\Delta t\left[f_3\bar{v}^{-\lambda\phi} - \frac{C_{pd}}{\bar{r}^{\lambda}\cos\phi}\left(\bar{\theta}_v^{r\lambda}\delta_\lambda\Pi - \overline{\theta_v\delta_r\Pi}^{r\lambda}\delta_\lambda r\right)\right] + \alpha_4\Delta t f_2\bar{w}^{r\lambda}\right\}^n \\ &+\left\{\begin{array}{l} u+(1-\alpha_3)\Delta t\left[f_3\bar{v}^{-\lambda\phi} - \frac{C_{pd}}{\bar{r}^{\lambda}\cos\phi}\left(\bar{\theta}_v^{r\lambda}\delta_\lambda\Pi - \overline{\theta_v\delta_r\Pi}^{r\lambda}\delta_\lambda r\right)\right] \\ -(1-\alpha_4)\Delta t f_2\bar{w}^{r\lambda}\end{array}\right\}_d^n\end{aligned} \tag{4.3.54}$$

2.) First 'Physics' corrector

S^u is written as the sum of two terms $S^u = S_1^u + S_2^u$ and we let the value of the physics time-weight, α_p, associated with S_1^u be 0 and that associated with S_2^u be 1. The terms for S_1^u are evaluated as functions of the model state at the previous, n th time-step. Therefore,

$$S_1^u = S_1^u\left(\{u^n\}\right) = G^u\left(\{u^n\}\right) \tag{4.3.55}$$

The effects of sub-grid scale gravity-wave drag. Let $\tilde{u}^{(p_1)}$ be the first physics predictor for u^{n+1}. This can be written as the sum of the first predictor $\tilde{u}^{(1)}$ and a first physics corrector as

$$\tilde{u}^{(p_1)} = \tilde{u}^{(1)} + \left(\tilde{u}^{(p_1)} - \tilde{u}^{(1)} \right) \tag{4.3.56}$$

The first physics corrector is defined as

$$\left(\tilde{u}^{(p_1)} - \tilde{u}^{(1)} \right) = \Delta t \left[S_1^u \right]_d^n \tag{4.3.57}$$

Let $R_u^{p_1} \equiv \tilde{u}^{(p_1)} - u^n$. This step is equivalently written as follows

$$R_u^{p_1} = R_u + \Delta t \left[S_1^u \right]_d^n \tag{4.3.58}$$

3.) Second 'Physics' corrector

S_2^n is then evaluated implicitly using model variables at time level $n+1$. Let $\tilde{u}^{(p_2)}$ be the second physics predictor for u^{n+1}. This can be written as the sum of the (first physics) predictor $\tilde{u}^{(p_1)}$ and a second physics corrector as

$$\tilde{u}^{(p_2)} = \tilde{u}^{(p_1)} + \left(\tilde{u}^{(p_2)} - \tilde{u}^{(p_1)} \right) \tag{4.3.59}$$

$$\left(\tilde{u}^{(p_2)} - \tilde{u}^{(p_1)} \right) = \Delta t \left[S_2^u \right]^* \tag{4.3.60}$$

$\left[S_2^u \right]^*$ is calculated from intermediate, unbalanced model state.

This step can briefly be represented thusly:

$$R_u^{p_2} = R_u^{p_1} + \Delta t \left[S_2^u \right]^* \tag{4.3.61}$$

Here, $R_u^{P_2} \equiv \tilde{u}^{(p_2)} - u^n$.

4.) First 'Dynamics' Corrector

Let $\tilde{u}^{(2)}$ be the a second dynamics predictor for u^{n+1}. This can be written as the sum of the second physics predictor $\tilde{u}^{(p_2)}$ and a first dynamics corrector as

$$\tilde{u}^{(2)} = \tilde{u}^{(p_2)} + \left(\tilde{u}^{(2)} - \tilde{u}^{(p_2)} \right) \tag{4.3.62}$$

This first dynamics corrector is defined as

$$R_u^+ = R_u^{p_2}\alpha_3\Delta t\left[\frac{C_{pd}}{\overline{r}^{\lambda}\cos\phi}\left(\overline{\left(\theta_v^*-\theta_v\right)}^{r\lambda}\delta_\lambda\Pi-\overline{\left(\theta_v^*-\theta_v\right)\delta_r\Pi}^{r\lambda}\delta_\lambda r\right)\right]^n \tag{4.3.63}$$

Here, $R_u^+ \equiv \tilde{u}^{(2)} - u^n$ and

$$\theta_v^* = \theta^*\left(\frac{1+\frac{1}{\varepsilon}m_v^*}{1+m_v^*+m_{cl}^*+m_{cf}^*}\right) \tag{4.3.64}$$

$m_X^* = \tilde{m}_X^{(p_2)}$, $X=(v,cl,cf)$, and θ^* are the latest predictors for m_X and θ at time $(n+1)\Delta t$.

5.) Second 'Dynamics' corrector

Let $\tilde{u}^{(3)}$ be a third dynamics predictor for u^{n+1}. This can be written as the sum of the second dynamics predictor $\tilde{u}^{(2)}$ and a second dynamics corrector as

$$\tilde{u}^{(3)} = \tilde{u}^{(2)} + \left(\tilde{u}^{(3)} - \tilde{u}^{(2)}\right) \tag{4.3.65}$$

This step is defined as

$$\left(\tilde{u}^{(3)} - \tilde{u}^{(2)}\right) = \alpha_3\Delta t\left[f_3\overline{v'}^{\lambda\phi} - \frac{C_{pd}}{\overline{r}^{\lambda}\cos\phi}\left(\overline{\theta_v^*}^{r\lambda}\delta_\lambda\Pi' - \overline{\theta_v^*\delta_r\Pi'}^{r\lambda}\delta_\lambda r\right)\right] \tag{4.3.66}$$

In this case

$$v' \equiv v^{n+1} - v^n, \Pi' \equiv \Pi^{n+1} - \Pi^n \tag{4.3.67}$$

Most conveniently,

$$R_u^{++} = R_u^+ - \alpha_3\Delta t\left[\frac{C_{pd}}{\overline{r}^{\lambda}\cos\phi}\left(\overline{\theta_v^*}^{r\lambda}\delta_\lambda\Pi' - \overline{\theta_v^*\delta_r\Pi'}^{r\lambda}\delta_\lambda r\right)\right] \tag{4.3.68}$$

6.) Third 'Dynamics' corrector

Let $\tilde{u}^{(4)}$ be the fourth dynamics predictor for u^{n+1}. This can be written as the sum of the third predictor $\tilde{u}^{(3)}$ and a third dynamics corrector.

$$u^{n+1} = \tilde{u}^{(3)} + \left(u^{n+1} - \tilde{u}^{(3)}\right) \tag{4.3.69}$$

This step is defined as follows

$$\left(u^{n+1}-\tilde{u}^{(3)}\right)=\frac{\alpha_3^2 f_3^2\Delta t^2}{1+\alpha_3^2 f_3^2\Delta t^2}\left(1-\bar{I}^{\lambda\lambda\phi\phi}\right)\left(u^n-\tilde{u}^{(3)}+\alpha_3 f_3\Delta t\overline{v'}^{\lambda\phi}\right) \tag{4.3.70}$$

I is a unit operator such that

$$\bar{I}^{\lambda\phi}F\equiv\overline{F}^{\lambda\phi} \tag{4.3.71}$$

This step is chiefly approximating the time tendency, v' as follows:

$$u'\equiv u^{n+1}-u^n=\alpha_3\Delta t f_3\overline{v'}^{\lambda\phi}+\left(\frac{I+\alpha_3^2 f_3^2\Delta t^2\bar{I}^{\lambda\lambda\phi\phi}}{1+\alpha_3^2 f_3^2\Delta t^2}\right)R_u^{++} \tag{4.3.72}$$

2. At levels $k=\frac{1}{2}$ ***and*** $k=N-\frac{1}{2}$

Discretisation of the u-component of the momentum equation at this points proceeds the same way as at the intermediate levels excepting the modification of certain terms on account of the presence of the upper boundaries.

(a.) At $k=\frac{1}{2}$

In calculation of $R_u\big|_{\eta_{\frac{1}{2}}}$, the term $\left(\overline{\theta_v^n}^{r\lambda}\delta_\lambda\Pi^n-\overline{\theta_v^n\delta_r\Pi^n}^{r\lambda}\delta_\lambda r\right)\Big|_{\eta_{\frac{1}{2}}}$ has to be evaluated, and both of the sub-terms involve an averaging over the layer $\left[\eta_0\equiv 0,\eta_1\right]$. Since θ_v, θ or q is not prognostically carried out at $\eta_0\equiv 0$, to circumvent the problem it is instead assumed that θ_v is constant in the layer $\left[\eta_0\equiv 0,\eta_1\right]$.

Therefore

$$\left(\theta_v\right)\big|_{\eta_0\equiv 0}=\left(\theta_v\right)\big|_{\eta_1} \tag{4.3.73}$$

And

$$\left(\overline{\theta_v^n}^{r\lambda}\delta_\lambda\Pi^n\right)\Big|_{\eta_{\frac{1}{2}}}=\left(\overline{\theta_v^n}^{\lambda}\right)\Big|_{\eta_1}\left(\delta_\lambda\Pi^n\right)\big|_{\eta_{\frac{1}{2}}} \tag{4.3.74}$$

The contribution due to the vertical derivative of Π normally spans two vertical mesh-lengths and this poses a problem in the second sub-term because data is not available below the surface. As a result, we evaluate $\left(\overline{\theta_v^n\delta_r\Pi^n}^{r\lambda}\right)\Big|_{\eta_{\frac{1}{2}}}$ at the bottom boundary as:

$$\theta_v^n\delta_r\Pi^n\big|_{\eta_0\equiv 0}=-\frac{g}{C_{pd}} \tag{4.3.75}$$

$\left(Ru\right)\big|_{\eta_{\frac{1}{2}}}$ is computed by evaluating the term $\left(f_2\overline{w}^{r\lambda}\right)\Big|_{\eta_{\frac{1}{2}}}$ by assuming that $w\big|_{\eta\equiv 0}=0$.

For $\left(R_u^+\right)\big|\eta_{\frac{1}{2}}$ and $\left(R_u^{++}\right)\big|_{\eta_{\frac{1}{2}}}$, the terms $\left[\overline{\left(\theta_v^*-\theta_v\right)}^{r\lambda}\delta_\lambda\Pi\right]\Big|_{\eta_{\frac{1}{2}}}$ and $\left[\overline{\theta_v^*}^{r\lambda}\delta_\lambda\Pi'\right]\Big|_{\eta_{\frac{1}{2}}}$ are evaluated as follows:

$$\left[\overline{\left(\theta_v^*-\theta_v\right)}^{r\lambda}\delta_\lambda\Pi\right]\Big|_{\eta_{\frac{1}{2}}}=\left(\overline{\left(\theta_v^*-\theta_v\right)}^{\lambda}\right)\Big|_{\eta_1}\left(\delta_\lambda\Pi\right)\big|_{\eta_{\frac{1}{2}}} \tag{4.3.76}$$

For the second term, an assumption is done as in (4.3.75). $\left[\overline{\left(\theta_v^*-\theta_v\right)\delta_r\Pi}^{r\lambda}\delta_\lambda r\right]\Big|_{\eta_{\frac{1}{2}}}$ is

evaluated by applying (4.3.75) with θ_v^* replacing θ_v^n.

Finally, the term $\left(R_u^{++}\right)\big|_{\eta_{\frac{1}{2}}}$ is computed as follows:

$$\left(\overline{\theta_v^*\delta_r\Pi'}^{r\lambda}\delta_\lambda r\right)\Big|_{\eta_{\frac{1}{2}}}=\left(\theta_v^*\right)\Big|_{\eta_1}\frac{\overline{\left(r\big|_{\eta_{\frac{1}{2}}}-r\big|_{\eta_0}\right)\left(\Pi'\big|_{\eta_{\frac{3}{2}}}-\Pi'\big|_{\eta_{\frac{1}{2}}}\right)}^{\lambda}}{\left(r\big|_{\eta_1}-r\big|_{\eta_0}\right)\left(r\big|_{\eta_{\frac{3}{2}}}-r\big|_{\eta_{\frac{1}{2}}}\right)}\left(\delta_\lambda r\right)\big|_{\eta_{\frac{1}{2}}} \tag{4.3.77}$$

Isentropic assumption has been made for θ_v in the layer $\left[\eta_0\equiv 0,\eta_1\right]$.

(b.) At $k=N-\dfrac{1}{2}$

We begin by computing $\left(Ru\right)\big|_{\eta_{N-\frac{1}{2}}}$. The term $\left(\overline{\theta_v^n}^{r\lambda}\delta_\lambda\Pi^n-\overline{\theta_v^n\delta_r\Pi^n}^{r\lambda}\delta_\lambda r\right)\Big|_{\eta_{N-\frac{1}{2}}}$ has to be

evaluated by averaging both of its sub-terms over the layer $\left[\eta_{N-1},\eta_N\equiv 1\right]$.

$\left(\overline{\theta_v^n\delta_r\Pi^n}^{r\lambda}\delta_\lambda r\right)\Big|_{\eta_{N-\frac{1}{2}}}$ has a difficulty arising from the fact that the vertical derivative of Π

normally spans two vertical mesh-lengths and data is unavailable above the rigid lid. This is overcome by defining the coordinate η in such a way as to make $r\big|_{\eta_{N-\frac{1}{2}}}=$ constant and so

$\left(\overline{\theta_v^n\delta_r\Pi^n}^{r\lambda}\delta_\lambda r\right)\Big|_{\eta_{N-\frac{1}{2}}}\equiv 0$ since $\left(\delta_\lambda r\right)\big|_{\eta_{N-\frac{1}{2}}}\equiv 0$.

Computing of $(Ru)\big|_{\eta_{N-\frac{1}{2}}}$ involves the use of $w\big|_{\eta_N}=0$ in evaluation of the term $\left(f_2\overline{w}^{r\lambda}\right)\Big|_{\eta_{N-\frac{1}{2}}}$.

Similarly, we evaluate $\left[\overline{\theta_v^*}^{r\lambda}\delta_\lambda\Pi'\right]\Big|_{\eta_{N-\frac{1}{2}}}$, $\left[\overline{\left(\theta_v^*-\theta_v^n\right)\delta_r\Pi^n}^{r\lambda}\delta_\lambda r\right]\Big|_{\eta_{N-\frac{1}{2}}}$, $\left[\overline{\left(\theta_v^*-\theta_v^n\right)}^{r\lambda}\delta_\lambda\Pi^n\right]\Big|_{\eta_{N-\frac{1}{2}}}$ and $\left[\overline{\theta_v^*\delta_r\Pi'}^{r\lambda}\delta_\lambda r\right]\Big|_{\eta_{N-\frac{1}{2}}}$ which will help in the calculation of $\left(Ru^+\right)\big|_{\eta_{N-\frac{1}{2}}}$ and $\left(Ru^{++}\right)\big|_{\eta_{N-\frac{1}{2}}}$.

4.3.5 Discretisation of the v-component of the momentum equation

1. At levels $k=\frac{1}{2},\frac{1}{3},...,N-\frac{1}{2}$

The V-component of momentum equation is given by

$$\frac{Dv}{Dt}+f_3u-f_1w+\frac{u^2\tan\phi}{r}+\frac{uw}{r}+\frac{C_{pd}\theta_v}{r}\left(\frac{\partial\Pi}{\partial\phi}-\frac{\partial\Pi}{\partial r}\frac{\partial r}{\partial\phi}\right)=S^v \tag{4.3.78}$$

Discretisation of this equation follows the same way as that in $u-$component of momentum equation.

The predictor-corrector stages are as follows. Let

$$R_v\equiv\overline{v}^{(1)}-v^n, R_v^{p_1}\equiv\overline{v}^{(p_1)}-v^n, R_v^{p_2}\equiv\overline{v}^{(p_2)}-v^n, R_v^{+}\equiv\overline{v}^{(2)}-v^n, R_v^{++}\equiv\overline{v}^{(3)}-v^n+\alpha_3 f_3\Delta t\overline{u'}^{\lambda\phi}$$

a. R_v is computed at the v points as follows:

$$\begin{aligned}R_v=&\left\{-v-\alpha_3\Delta t\left[f_3\overline{u}^{\lambda\phi}+\frac{C_{pd}}{\overline{r}^{\phi}}\left(\overline{\theta_v}^{r\phi}\delta_\phi\Pi-\overline{\theta_v\delta_r\Pi}^{r\phi}\delta_\phi r\right)\right]+\alpha_4\Delta t f_1\overline{w}^{r\phi}\right\}^n\\&+\left\{v-(1-\alpha_3)\Delta t\left[f_3\overline{u}^{\lambda\phi}+\frac{C_{pd}}{\overline{r}^{\phi}}\left(\overline{\theta_v}^{r\phi}\delta_\phi\Pi-\overline{\theta_v\delta_r\Pi}^{r\phi}\delta_\phi r\right)\right]+(1-\alpha_4)\Delta t f_1\overline{w}^{r\phi}\right\}_d^n\end{aligned} \tag{4.3.79}$$

b. $R_v^{p_1}$ is then computed at v points as follows:

$$R_v^{p_1}=R_v+\Delta t\left[S_1^v\right]_d^n \tag{4.3.80}$$

In this case, $\left[S_1^v\right]_d^n$ is the parallel or process-split component of the physics increment.

c. $R_v^{p_2}$ is equally calculated as follows:

$$R_v^{p_2} = R_v^{p_1} + \Delta t\left[S_2^v\right]^* \tag{4.3.81}$$

$\left[S_2^v\right]^*$ is the sequential, or time-split, component of the physics increment computed in the same way as $\left[S_1^v\right]_d^n$.

d. We now obtain R_v^+ at the v points.

$$R_v^+ = R_v^{p_2} - \alpha_3\Delta t\left[\frac{C_{pd}}{\overline{r}^{\phi}}\left(\overline{\left(\theta_v^* - \theta_v\right)}^{r\phi}\delta_\phi\prod - \overline{\left(\theta_v^* - \theta_v\right)\delta_r\prod}^{r\phi}\delta_\phi r\right)\right]^n, \tag{4.3.82}$$

Where

$$\theta_v^* = \theta^*\left(\frac{1+\frac{1}{\varepsilon}m_v^*}{1+m_v^*+m_{cl}^*+m_{cf}^*}\right) \tag{4.3.83}$$

e. R_v^{++} is obtained at v points as follows:

$$R_v^{++} = R_v^+ - \alpha_3\Delta t\left[\frac{C_{pd}}{\overline{r}^{\phi}}\left(\overline{\theta_v^*}^{r\theta}\delta_\phi\prod' - \overline{\theta_v^*\delta_r\prod'}^{r\phi}\delta_\phi r\right)\right] \tag{4.3.84}$$

f. Finally, the approximation of the tendency v' is obtained as follows

$$v' \equiv v^{n+1} - v^n = -\alpha_3\Delta t f_3\overline{u'}^{\lambda\phi} + \left(\frac{1+\alpha_3^2 f_3^2\Delta t^2\overline{I}^{\lambda\lambda\phi\phi}}{1+\alpha_3^2 f_3^2\Delta t^2}\right)R_v^{++} \tag{4.3.85}$$

In this case,

$$\begin{aligned}\prod' &\equiv \prod^{n+1} - \prod^n \\ v' &\equiv v^{n+1} - v^n\end{aligned} \tag{4.3.86}$$

4.3.6 Discretisation of the vertical component of the momentum equation

The vertical component of momentum equation amenable for weather prediction is given by:

$$\frac{Dw}{Dt} + \left(-f_2 u + f_1 v - \frac{u^2+v^2}{r}\right) + g + C_{pd}\theta_v\frac{\partial\prod}{\partial r} = 0 \tag{4.3.87}$$

At levels $k = 1, 2, ..., N-1$

Using the two-time-level off-centred semi-implicit semi-Lagrangian scheme, the following is obtained at points $\left(\lambda_{I-\frac{1}{2}}, \phi_{J-\frac{1}{2}}, \eta_k\right)$:

$$\frac{w^{n+1}-w_d^n}{\Delta t}=\alpha_4\left[\left(f_2\bar{u}^{\lambda r}-f_1\bar{v}^{\phi r}\right)-g-C_{pd}\theta_v\delta_r\Pi\right]^{n+1}$$
$$+\left(1-\alpha_4\right)\left[\left(f_2\bar{u}^{\lambda r}-f_1\bar{v}^{\phi r}\right)-g-C_{pd}\theta_v\delta_r\Pi\right]_d^n \tag{4.3.88}$$

Due to the complexity of treating the Coriolis terms and non-linear pressure gradient term, the following predictor corrector method is useful.

1. Predictor

Suppose $\tilde{w}^{\left(\frac{1}{2}\right)}$ is the first predictor for w^{n+1}. Thus

$$\frac{\tilde{w}^{(1)}-w_d^n}{\Delta t}=\alpha_4\left[\left(f_2\bar{u}^{\lambda r}-f_1\bar{v}^{\phi r}\right)-g-C_{pd}\theta_v\delta_r\Pi\right]^{n}$$
$$+\left(1-\alpha_4\right)\left[\left(f_2\bar{u}^{\lambda r}-f_1\bar{v}^{\phi r}\right)-g-C_{pd}\theta_v\delta_r\Pi\right]_d^n \tag{4.3.89}$$

Equivalently, let $R_w\equiv\tilde{w}^{(1)}-w^n$. This step is ideally the computation of R_w as follows:

$$R_w=w_d^n-w^n+\alpha_4\left[\left(f_2\bar{u}^{\lambda r}-f_1\bar{v}^{\phi r}\right)-g-C_{pd}\theta_v\delta_r\Pi\right]^{n}$$
$$+\left(1-\alpha_4\right)\left[\left(f_2\bar{u}^{\lambda r}-f_1\bar{v}^{\phi r}\right)-g-C_{pd}\theta_v\delta_r\Pi\right]_d^n \tag{4.3.90}$$

2. First corrector

Here we let $\tilde{w}^{(2)}$ be a second predictor for w^{n+1}. This can be written as the sum of the first predictor and the first corrector, that is,

$$\tilde{w}^{(2)}=\tilde{w}^{(1)}+\left(\tilde{w}^{(2)}-\tilde{w}^{(1)}\right) \tag{4.3.91}$$

The first corrector is defined as follows:

$$\left(\tilde{w}^{(2)}-\tilde{w}^{(1)}\right)=-\alpha_4\Delta t\left[C_{pd}\left(\theta_v^*-\theta_v^n\right)\delta_r\Pi^n\right] \tag{4.3.92}$$

Where

$$\theta_v^*=\theta^*\left(\frac{1+\frac{1}{\varepsilon}m_v^*}{1+m_v^*+m_{cl}^*+m_{cf}^*}\right) \tag{4.3.93}$$

Most precisely, this step can be written as follows:

Let $R_w^+=\tilde{w}^{(2)}-w^n$, we compute R_w^+

$$R_w^+ = R_w - \alpha_4 \Delta t C_p \left(\theta_v^* - \theta_v^n\right)\delta_r \Pi^n \tag{4.3.94}$$

3. Second corrector

Let $\tilde{w}^{(2)}$ be a third predictor for w^{n+1}. This can be written as the sum of the second predictor and the second corrector.

$$\tilde{w}^{(3)} = \tilde{w}^{(2)} + \left(\tilde{w}^{(3)} - \tilde{w}^{(2)}\right) \tag{4.3.95}$$

We define the second corrector as follows:

$$\left(\tilde{w}^{(3)} - \tilde{w}^{(2)}\right) = -\alpha_4 \Delta t C_{pd} \theta_v^* \delta_r \Pi' \tag{4.3.96}$$

Here, $\Pi' \equiv \Pi^{n+1} - \Pi^n$. This leads to an implicit coupling of the momentum equation with the other governing equations and eventually leads to a Helmholtz to be solved for the Exner pressure tendency, Π'.

4. Third Corrector

Let $\tilde{w}^{(4)} \equiv w^{n+1}$ be the fourth and final predictor, which can be written as the sum of the third predictor and the third corrector.

$$w^{n+1} = \tilde{w}^{(3)} + \left(w^{n+1} - \tilde{w}^{(3)}\right) \tag{4.3.97}$$

The third corrector is defined as

$$\left(w^{n+1} - \tilde{w}^{(3)}\right) = -\alpha_4 \Delta t C_{pd} \left(\theta_v^{n+1} - \theta_v^*\right)\delta_r \Pi^n \tag{4.3.98}$$

The final discretisation of the $w-$component of the momentum equation can be written as

$$\begin{aligned} &\frac{w^{n+1} - w_d^n}{\Delta t} = \alpha_4 \left[\left(f_2 \bar{u}^{\lambda r} - f_1 \bar{v}^{\phi r}\right) - g\right]^n \\ &-\alpha_4 \left[C_{pd}\theta_v^{n+1}\delta_r \Pi^{n+1} - C_{pd}\left(\theta_v^{n+1} - \theta_v^*\right)\left(\delta_r \Pi^{n+1} - \delta_r \Pi^n\right)\right] \\ &+\left(1-\alpha_4\right)\left[\left(f_2 \bar{u}^{\lambda r} - f_1 \bar{v}^{\phi r}\right) - g - C_{pd}\theta_v \delta_r \Pi\right]_d^n \end{aligned} \tag{4.3.99}$$

4.3.7 Discretisation of the continuity equation

The momentum equation can be written in Eulerian flux form as follows:

$$\frac{\partial}{\partial t}\left(r^2\rho_y\frac{\partial r}{\partial\eta}\right)+\left[\frac{1}{\cos\phi}\frac{\partial}{\partial\lambda}\left(r^2\rho_y\frac{\partial r}{\partial\eta}\frac{u}{r}\right)+\frac{1}{\cos\phi}\frac{\partial}{\partial\phi}\left(r^2\rho_y\frac{\partial r}{\partial\eta}\frac{v\cos\phi}{r}\right)\right]$$
$$+\frac{\partial}{\partial\eta}\left(r^2\rho_y\frac{\partial r}{\partial\eta}\dot{\eta}\right)=0 \qquad (4.3.100)$$

Where

$$\frac{\partial r}{\partial\eta}\dot{\eta}=w-\frac{u}{r\cos\phi}\frac{\partial r}{\partial\lambda}-\frac{v}{r}\frac{\partial r}{\partial\phi}, \qquad (4.3.101)$$

Furthermore,

$$\dot{\eta}\big|_{\eta=0}=\dot{\eta}\big|_{\eta=1}=0 \qquad (4.3.102)$$

By use of (1.24) we can write (1.23) as

$$\frac{\partial}{\partial t}\left(r^2\rho_y\frac{\partial r}{\partial\eta}\right)+\left\{\begin{array}{l}\frac{1}{\cos\phi}\frac{\partial}{\partial\lambda}\left(r^2\rho_y\frac{\partial r}{\partial\eta}\frac{u}{r}\right)+\frac{1}{\cos\phi}\frac{\partial}{\partial\phi}\left(r^2\rho_y\frac{\partial r}{\partial\eta}\frac{v\cos\phi}{r}\right)\\ -\frac{\partial}{\partial\eta}\left(r^2\rho_y\frac{u}{r\cos\phi}\frac{\partial r}{\partial\lambda}+r^2\rho_y\frac{v}{r}\frac{\partial r}{\partial\phi}\right)+\frac{\partial}{\partial\eta}\left(r^2\rho_y w\right)\end{array}\right\}=0 \qquad (4.3.103)$$

At levels $k=\frac{1}{2},\frac{3}{2},...,N-\frac{1}{2}$

By a two-time-level off-centred semi-implicit Eulerian scheme, the following is obtained at the point $p\left(\lambda_{I-\frac{1}{2}},\phi_{J-\frac{1}{2}},\eta_{K-\frac{1}{2}}\right)$,

$$\frac{\left(r^2\rho_y\right)^{n+1}-\left(r^2\rho_y\right)^{n}}{\Delta t}=-\frac{1}{\delta_\eta r}\left[\begin{array}{l}\overline{\frac{1}{\cos\phi}\delta_\lambda\left(\frac{\overline{r^2\rho_y\delta_\eta r}^{\lambda}}{\bar{r}^{\lambda}}u\right)}^{\alpha_1}\\ +\overline{\frac{1}{\cos\phi}\delta_\phi\left(\frac{\overline{r^2\rho_y\delta_\eta r}^{\phi}}{\bar{r}^{\phi}}v\cos\phi\right)}^{\alpha_1}\\ +\overline{\delta_\eta\left(\overline{r^2\rho_y}^{r}\dot{\eta}\delta_\eta r\right)}^{average}\end{array}\right] \qquad (4.3.104)$$

$\frac{\partial r}{\partial\eta}$ is time independent and time-weighting is done at mesh-point at times $n\Delta t$ and $(n+1)\Delta t$.

1. Predictor

Let $\tilde{p}_y^{(1)}$ be a predictor for $p_y^{(n+1)}$. This is meant to replace all the terms evaluated as time averages of quantities at mesh-points at time levels $n\Delta t$ and $(n+1)\Delta t$ by their values at the same mesh-points but at time $n\Delta t$. Therefore,

$$\frac{\left(r^2\tilde{\rho}_y^{(1)}\right)-\left(r^2\rho_y^n\right)}{\Delta t}=-\frac{1}{\delta_\eta r}\left[\begin{array}{l}\dfrac{1}{\cos\phi}\delta_\lambda\left(\dfrac{\overline{r^2\rho_y^n\delta_\eta r}^{\lambda}}{\bar{r}^{\lambda}}\,u^n\right)\\ +\dfrac{1}{\cos\phi}\delta_\phi\left(\dfrac{\overline{r^2\rho_y^n\delta_\eta r}^{\phi}}{\bar{r}^{\phi}}\,v^n\cos\phi\right)\\ +\delta_\eta\left(\overline{r^2\rho_y^n}^{r}\dot{\eta}^n\delta_\eta r\right)\end{array}\right] \tag{4.3.105}$$

In this case,

$$\dot{\eta}^n=\frac{1}{\delta_\eta r}\left(w^n-\frac{\overline{\bar{u}^n}^{\lambda}}{\bar{r}^{\lambda}\cos\phi}\delta_\lambda r\;-\frac{\overline{\bar{v}^n}^{\phi}}{\bar{r}^{\phi}}\delta_\phi r\right)^n \tag{4.3.106}$$

And

$$\dot{\eta}^n\Big|_{\eta_0\equiv 0}=\dot{\eta}^n\Big|_{\eta_N\equiv 1}=0 \tag{4.3.107}$$

2. Corrector

ρ_y^{n+1} can be written as the sum of the predictor and corrector as follows:

$$\rho_v^{n+1}=\tilde{\rho}_y^{(1)}+\left(\rho_y^{n+1}-\tilde{\rho}_y^{(1)}\right) \tag{4.3.108}$$

This corrector is defined by

$$\left(r^2\rho_y^{n+1}\right)-\left(r^2\tilde{\rho}_y^{(1)}\right)=-\frac{\Delta t}{\delta_\eta r}\left\{\begin{array}{l}\dfrac{1}{\cos\phi}\delta_\lambda\left(\dfrac{\overline{r^2\rho_y^n\delta_\eta r}^{\lambda}}{\bar{r}^{\lambda}}\,\alpha_1 u'\right)\\ +\dfrac{1}{\cos\phi}\delta_\phi\left(\dfrac{\overline{r^2\rho_y^n\delta_\eta r}^{\phi}}{\bar{r}^{\phi}}\,\alpha_1 v'\cos\phi\right)\\ +\delta_\eta\left[\overline{r^2\rho_y^n}^{r}\left(\bar{\dot{\eta}}^{average}-\dot{\eta}^n\right)\delta_\eta r\right]\end{array}\right\} \tag{4.3.109}$$

In this case,

$$\dot{\eta}^{average}=\frac{1}{\delta_\eta r}\left[\bar{w}^{\alpha_2}-\overline{\left(\frac{\overline{\bar{u}^{\eta}}^{\lambda}}{\bar{r}^{\lambda}\cos\phi}\delta_\lambda r+\frac{\overline{\bar{v}^{\eta}}^{\phi}}{\bar{r}^{\phi}}\delta_\phi r\right)}^{\alpha_1}\right] \tag{4.3.110}$$

$$\dot{\eta}^{average}\Big|_{\eta_0\equiv 0}=\bar{\dot{\eta}}^{average}\Big|_{\eta_N\equiv 1}=0 \tag{4.3.111}$$

And

$$u'\equiv u^{n+1}-u^n;v'\equiv v^{n+1}-v^n \tag{4.3.112}$$

4.3.8. Discretisation of thermodynamic equation, moisture equations and equation of state

These equations are discretised using the SISL scheme following the same way as the momentum equation. The various predictor corrector steps are shown in section 11 of Staniforth *et al.* (2004).

4.3.8.1. Discretised form of Thermodynamic Equation

When discretisation is done at levels $k=1,2,...,N-1,$ it yields:

$$\theta'=(\theta^*-\theta^n)-\alpha_2\Delta t(w'\delta_{2r}\theta_{ref}) \tag{4.3.113}$$

$\theta'\equiv\theta^{n+1}-\theta^n$ and $\theta^*\equiv\tilde{\theta}^{(p_2)}$ is the latest available predictor for θ at time $(n+1)\Delta t$.

At levels $k=0$ and $k=N,$ respectively,

$$\theta'\big|_{\eta_0\equiv 0}=\theta'\big|_{\eta_1} \tag{4.3.114}$$

And

$$\theta'\big|_{\eta_{N=1}} = \left(\theta^* - \theta^n\right)\Big|_{\eta_{N=1}} \tag{4.3.115}$$

4.3.8.2. Discretised forms of Moisture Equations

The final forms at levels $k = 1, 2, ..., N$ are:

$$m_X^* \equiv \widetilde{m}_X^{(p_2)};\ X = (v, cl.cf) \tag{4.3.116}$$

At level $k = 0$, we have the following:

$$\left(m_X^*\right)\Big|_{\eta_0 \equiv 0} = \left(m_X^*\right)\Big|_{\eta_1},\ X = (v, cl.cf) \tag{4.3.117}$$

4.3.8.3. Discretised form of the Equation of State

After linearising and discretising the equation of state, the final form at levels $k = \frac{1}{2}, \frac{3}{2}, ..., N - \frac{1}{2}$ is:

$$k_d \prod^n \overline{\theta_v^n}^r \rho' + \left(k_d \overline{\theta_v^n}^r \rho^n - \frac{\rho^n}{R_d \prod^n} \right) \prod' + k_d \prod^n \rho^n \overline{\theta_v'}^r = \frac{\rho^n}{C_{pd}} - k_d \prod^n \rho^n \overline{\theta_v^n}^r \tag{4.3.118}$$

In this case: $\theta_v' \equiv \theta_v^{n+1} - \theta_v^n,\ \rho' \equiv \rho^{n+1} - \rho^n$.

So far, we have had an overview of the discretisation of the governing equations having been cast in a form amenable to practical implementation on a spherical surface. In this overview, we have not shown the treatment at the poles partly because that would make the task enormous, and partly because we aimed at concerning ourselves with the numerical methods used and their analysis. Suffice it to indicate that the prognostic variables are mostly held constant at the poles.

The discretisation of governing equations yields $13N + 7$ levels of unknown with 13 variables. 8 of these variables are determined from the associated prognostic equation of the variable. These variables are $u,\ v,\ w,\ \rho_y,\ \theta,\ m_v,\ m_{cl}$ and m_{cf}. The other 5; $\dot{\eta},\ \rho,\ \theta_v,\ \prod$ and p are diagnostically determined. To solve this coupled set of linear equations, an equivalent Helmholtz equation is obtained by decomposing them algebraically. To crown this chapter, we shall state the Helmholtz equation as indicated in Buckeridge (2010). The derivation of the Helmholtz equation has been done and the explanation can be obtained in Davies *et al.* (2004).

4.4. The Helmholtz Equation

Algebraic decomposition of discretised equations yields an elliptic eqaution of Helmholtz type. Due to cross derivative terms in this equation, it is difficult to implement it. Simplification of this equation is done by rendering the Coriolis terms constant. With these simplifications, we get the following Helmholtz equation that is amenable to implementation:

$$-k\Pi^n\theta^n\Delta t^2\alpha^2 C_p\rho^n\theta^{(1)}A\left\{\begin{array}{l}\dfrac{1}{r^2\sin^2\phi}\dfrac{\partial^2\Pi'}{\partial\lambda^2}+\dfrac{1}{r^2\sin\theta}\dfrac{\partial}{\partial\phi}\left(\sin\phi\dfrac{\partial\Pi'}{\partial\phi}\right)+\\ \dfrac{1}{r^2}\dfrac{\partial}{\partial r}\left(\dfrac{r^2}{GA}\dfrac{\partial\Pi'}{\partial r}\right)\end{array}\right\}$$

$$-\left(k\Pi^n\Delta t^2\alpha^2 C_p\rho^n\theta^{(1)}\frac{\partial\theta^{(1)}}{\partial rG}\right)\frac{\partial\Pi'}{\partial r}+\left(\frac{p^n}{kC_p\Pi^n}-k\theta^n\rho^n\right)\Pi' \qquad (4.4.1)$$

$$=k\Pi^n\theta^n\rho^n-\frac{p^n}{C_p}+\Psi^n$$

As indicated in Buckeridge (2010), (4.4.1) is solved with the assumption that the Earth's surface and the top of the atmosphere are rigid boundaries. We therefore impose rigid boundary conditions $\prod'.\mathbf{n}=0$ at these boundaries. $\vec{n}$ is a unit vector pointing outwards from the boundary. This leads to the following Neumann boundary condition:

$$\frac{\partial\Pi'}{\partial n}=-\mathbf{x}.\mathbf{n} \qquad (4.4.2)$$

The Helmoltz equation (4.4.1) is solved for Π', which is used to calculate the Exner pressure in the next time-step. The Exner Pressure thus obtained is used to find u^{n+1}, v^{n+1} and w^{n+1} which are afterwards used to obtain θ^{n+1} and ρ^{n+1}.

For the purpose of our analysis in the subsequent parts, we are going to use a 2-dimensional Helmholtz equation shown below.

$$\nabla^2\prod'(x,y)+\omega^2\prod'(x,y)=\Phi(x,y) \qquad (4.4.3)$$

$\nabla^2=\dfrac{\partial^2}{\partial x^2}+\dfrac{\partial^2}{\partial x^2}$ is the Laplacian in rectangular coordinates, ω is the wave number, $\prod'$ is the increment in pressure field and Φ represent the value of the pressure field calculated from the previous time step.

In the next chapter, we are going to analyse the consistency, stability and convergence of the Semi-implicit Semi-Lagrangian scheme that has been used in this chapter for discretisation.

CHAPTER 5: CONSISTENCY, CONVERGENCE AND STABILITY ANALYSIS OF THE SEMI-IMPLICIT SEMI-LAGRANGIAN DISCRETISATION SCHEME

5.1 Consistency

A finite difference scheme is said to be consistent if the scheme reduces to the original differential equation as the time step and the space steps in the independent variables vanish. As has been pointed out in Urroz(2004), any finite difference scheme based on reasonable approximation of the derivatives should be consistent. The Semi-implicit semi-Lagrangian time discretisation scheme given by

$$\frac{DF}{Dt} = \Psi \equiv \frac{F^{n+1} - F_d^n}{\Delta t} = \overline{\Psi}, \tag{5.1.1}$$

Here, $\frac{D}{Dt} = \frac{\partial}{\partial t} + \mathbf{u}.\nabla$.

In this case, the finite difference shown in (5.1.1) will reproduce the original equation as $\Delta t \to 0$ and $\Delta x \to 0$ (we have considered x as the only special variable for this analysis).

$$\frac{F_i^{n+1} - F_i^n}{\Delta t} = \left(\frac{\partial}{\partial t} F(x,t)\right) + o(\Delta t) \tag{5.1.2}$$

And

$$u\frac{F_{i+1}^n - F_{i-1}^n}{2\Delta x} = u\frac{\partial}{\partial x} F(x,t) + u\left[o\left(\Delta x^2\right)\right] \tag{5.1.3}$$

Thus (5.1.1) can be written as

$$\left(\frac{\partial}{\partial t} F(x,t)\right) + o(\Delta t) + u\left(\frac{\partial}{\partial x} F(x,t)\right) + u\left[o\left(\Delta x^2\right)\right] = \Psi \tag{5.1.4}$$

As $\Delta x \to 0$ and $\Delta t \to 0$, The scheme reduces to the original equation.

$$\left(\frac{\partial}{\partial t} F(x,t)\right) + u\left(\frac{\partial}{\partial x} F(x,t)\right) = \Psi \tag{5.1.5}$$

Or in vector form

$$\frac{DF}{Dt} = \Psi \tag{5.1.6}$$

Hence the SISL scheme is consistent.

5.2 Stability analysis of Semi-Lagrangian Semi-implicit scheme

It is enough to show that for any initial perturbation, the solution to governing equations remains bounded. This means that the eigenvalues of the discrete propagator lie within the unit circle.

We shall begin by stating two theorems whose proofs are found in appendix A of Payne (2008).

Theorem 1: *Let*

$$\mathbf{L}_4 = \begin{bmatrix} 0 & -a & -b & 0 \\ a & 0 & 0 & 0 \\ b & 0 & 0 & -c \\ 0 & 0 & c & 0 \end{bmatrix} \tag{5.2.1}$$

$$\mathbf{L}_4^{\alpha} = \begin{bmatrix} 0 & -\alpha_1 a & -\alpha_2 b & 0 \\ \alpha_3 a & 0 & 0 & 0 \\ \alpha_4 b & 0 & 0 & -\alpha_4 c \\ 0 & 0 & \alpha_2 c & 0 \end{bmatrix} \tag{5.2.2}$$

Where a, b, c *and* α_i *are real numbers. If* $\alpha_1\alpha_3 \geq 0$ *and* $\alpha_2\alpha_4 \geq 0$, *then* $\mathbf{I} + \mathbf{L}_4^{\alpha}$ *is non-singular and the matrix* $\mathbf{M}_4 = \mathbf{I} - \left(\mathbf{I} + \mathbf{L}_4^{\alpha}\right)^{-1}\mathbf{L}_4$ *exists. Then if* $ac \neq 0$ *and* $\alpha_1, \alpha_2, \alpha_3, \alpha_4 > \frac{1}{2}$, *the eigenvalues of* $\mathbf{M}_4$ *all lie strictly inside the unit circle.*

Theorem 2: *Let*

$$\mathbf{L}_5 = \begin{bmatrix} 0 & -a & -d & -b & 0 \\ a & 0 & -f & -f & 0 \\ d & f & 0 & 0 & 0 \\ b & 0 & 0 & 0 & -c \\ 0 & 0 & 0 & c & 0 \end{bmatrix} \tag{5.2.3}$$

$$\mathbf{L}_5^{\alpha} = \begin{bmatrix} 0 & -\alpha_1 a & -\alpha_1 d & -\alpha_2 b & 0 \\ \alpha_3 a & 0 & -\alpha_3 f & 0 & 0 \\ \alpha_3 d & \alpha_3 f & 0 & 0 & 0 \\ \alpha_4 b & 0 & 0 & 0 & -\alpha_4 c \\ 0 & 0 & 0 & \alpha_2 c & 0 \end{bmatrix} \tag{5.2.4}$$

Where a,b,c,d, f *and* α_i *are real. If* $\alpha_1\alpha_3 \geq 0$ *and* $\alpha_2\alpha_4 \geq 0$, *then* $\mathbf{I}+\mathbf{L}_5^{\alpha}$ *is non-singular and the matrix*

$$\mathbf{M}_5 = \mathbf{I} - \left(\mathbf{I}+\mathbf{L}_5^{\alpha}\right)^{-1}\mathbf{L}_5 \tag{5.2.5}$$

Exists. Then if $ac \neq 0$ *and*

Then one of the eigenvalues of $\mathbf{M}_5$ *is equal to 1 and the other four all lie within the unit circle.*

Example 1: for Theorem 1

Let $a=b=c=1$ and $\alpha_1=\alpha_2=\alpha_3=\alpha_4=0.8$. Clearly $\alpha_1\alpha_3 \geq 0, \alpha_2\alpha_4 \geq 0$ and $ac \neq 0$. Furthermore, $\alpha_1,\alpha_2,\alpha_3,\alpha_4 > \frac{1}{2}$.

In the following Matlab output $\mathbf{L}=\mathbf{L}_4$, $\mathbf{N}=\mathbf{L}_4^{\alpha}$ and $\mathbf{M}=\mathbf{M}_4$.

```
>> L=[0 -1 -1 0;1 0 0 0;1 0 0 -1;0 0 1 0]

L =

     0    -1    -1     0
     1     0     0     0
     1     0     0    -1
     0     0     1     0

>> N=[0 -0.8 -0.8 0;0.8 0 0 0;0.8 0 0 -0.8;0 0 0.8 0]

N =

         0   -0.8000   -0.8000         0
    0.8000         0         0         0
    0.8000         0         0   -0.8000
         0         0    0.8000         0

>> M=eye(4)-((eye(4)+N)^-1)*L

M =

    0.3657    0.4926    0.3003    0.2403
   -0.4926    0.6060   -0.2403   -0.1922
   -0.3003   -0.2403    0.3657    0.4926
    0.2403    0.1922   -0.4926    0.6060

>> Lambda=eig(M)

Lambda =

   0.2172 + 0.6048i
   0.2172 - 0.6048i
   0.7545 + 0.4966i
   0.7545 - 0.4966i
```

It is enough to show that $r_i < 1$. In this example, $r_1 = r_2 = 0.6426$ and $r_3 = r_4 = 0.9033$. Therefore, $r_1, r_2, r_3, r_4 < 1$.

Example 2: for Theorem 2

```
>> L=[0 -1 -1 -1 0;1 0 -1 0 0;1 1 0 0 0;1 0 0 0 -1;0 0 0 1 0]

L =

     0    -1    -1    -1     0
     1     0    -1     0     0
     1     1     0     0     0
     1     0     0     0    -1
     0     0     0     1     0

>> N=[0 -0.8 -0.8 -0.8 0;0.8 0 -0.8 0 0;0.8 0.8 0 0 0;0.8 0 0 0 -0.8;0 0 0 0.8 0]

N =

         0   -0.8000   -0.8000   -0.8000         0
    0.8000         0   -0.8000         0         0
    0.8000    0.8000         0         0         0
    0.8000         0         0         0   -0.8000
         0         0         0    0.8000         0

>> M=eye(5)-((eye(5)+N)^-1)*L

M =

    0.3258    0.0562    0.5056    0.2809    0.2247
   -0.5056    0.4629    0.1658   -0.2466   -0.1973
   -0.0562   -0.6152    0.4629   -0.0274   -0.0219
   -0.2809   -0.0274   -0.2466    0.3752    0.5001
    0.2247    0.0219    0.1973   -0.5001    0.5999

>> Lambda=eig(M)

Lambda =

   0.1011 + 0.5618i
   0.1011 - 0.5618i
   1.0000
   0.5122 + 0.6098i
   0.5122 - 0.6098i
```

Clearly, one eigenvalue is equal to 1 and the other four all lie inside the unit circle.

We now use these developments to illustrate the stability of SISL scheme.

Proof of stability of SISL scheme

The continuous governing equations, in Cartesian coordinates without forcing terms are given by:

$$\frac{Du}{Dt} = fv - C_p\theta\frac{\partial \Pi}{\partial x} \tag{5.2.6}$$

$$\frac{Dv}{Dt} = -fu - C_p\theta\frac{\partial \Pi}{\partial y} \tag{5.2.7}$$

$$\frac{Dw}{Dt} = -g - C_p\theta\frac{\partial \Pi}{\partial z} \tag{5.2.8}$$

$$\frac{D\theta}{Dt} = 0 \tag{5.2.9}$$

$$\frac{Dp}{Dt} = -\rho\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right) \tag{5.2.10}$$

$$\Pi^{\frac{k-1}{k}}\ \rho\theta = \frac{p_0}{kC_p} \tag{5.2.11}$$

In this case, $\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + \vec{u}.\vec{\nabla}$.

To begin with, we represent each independent variable $F(x, y, z, t)$ as the sum of the motionless horizontally-independent basic state $F_s(z)$, in which

$$\begin{aligned} u_s &= v_s = w_s = 0 \\ \theta_s &= \theta_s(z) \\ \rho_s &= \rho_s(z) \\ \Pi_s &= \Pi_s(z) \end{aligned} \tag{5.2.12}$$

Let $F'(x, y, z, t)$ be a perturbation and assume that the basic state satisfies the governing equations, and products of perturbation are neglected. This implies that the basic state is independent of (x, y, z), and is hydrostatic:

$$-g - C_p\theta_s\frac{\partial \Pi_s}{\partial z} = 0$$

Let the basic state be at the same temperature, T_s. Then,

$$\begin{aligned} \theta_s(z) &= T_s e^{kaz} \\ \Pi_s(z) &= e^{-kaz} \\ \rho_s(z) &= \rho_0 e^{-az} \end{aligned} \tag{5.2.13}$$

Here, $a = \frac{g}{RT_s}$ and $\rho_0 = \frac{p_0}{RT_s}$.

In the following equations, we substitute $F(x,y,z,t) = F_s(z) + F'(x,y,z,t)$ into the SISL discretisation of equations (5.2.13)-(5.2.17) and use a linearised equation of state (5.2.17) to eliminate ρ. The time-discrete spatially-continuous equations for the perturbation are:

$$\frac{1-k}{k\prod_s}\frac{D^L \prod'}{Dt} = -\left(\frac{1}{\theta_s}\frac{\partial \theta_s}{\partial z} + \frac{1}{\rho_s}\frac{\partial \rho_s}{\partial z}\right)\overline{w'}^{\alpha_2} - \frac{\partial \overline{u'}^{\alpha_1}}{\partial x} - \frac{\partial \overline{v'}^{\alpha_1}}{\partial y} - \frac{\partial \overline{w'}^{\alpha_2}}{\partial z} \tag{5.2.14}$$

$$\frac{D^L u'}{Dt} = -C_p \theta_s \frac{\partial \overline{\prod'}^{\alpha_3}}{\partial x} + f\overline{v'}^{\alpha_3} \tag{5.2.15}$$

$$\frac{D^L v'}{Dt} = -C_p \theta_s \frac{\partial \overline{\prod'}^{\alpha_3}}{\partial y} - f\overline{u'}^{\alpha_3} \tag{5.2.16}$$

$$\frac{D^L w'}{Dt} = -C_p \theta_s \frac{\partial \overline{\prod'}^{\alpha_4}}{\partial z} - C_p \frac{\partial \prod_s}{\partial z}\overline{\theta'}^{\alpha_4} \tag{5.2.17}$$

$$\frac{D^L \theta'}{Dt} = -\frac{\partial \theta_s}{\partial z}\overline{w'}^{\alpha_2} \tag{5.2.18}$$

It should be noted that

$$\frac{D^L F}{Dt} = \frac{F(N+1) - F_d(N)}{\Delta t} \tag{5.2.19}$$

$$\overline{F}^{\alpha_i} \equiv (1-\alpha_i)F_d(N) + \alpha_i F(N+1) \tag{5.2.20}$$

$F_d = F$ since $\tilde{u}_s = \tilde{0}$.

Let the domain be

$$\begin{aligned} 0 \le x \le L \\ 0 \le y \le J \\ 0 \le z \le H \end{aligned} \tag{5.2.21}$$

We then consider the boundaries. Let there be a rigid boundary at $z = 0$ and $z = H$, and periodicity in the lateral direction so that all perturbation variables are proportional to

$$\exp\left(i\left(m'x + l'y\right)\right) \tag{5.2.22}$$

Here, $m' = \frac{2\pi m}{L}$ and $L' = \frac{2\pi l}{J}$ for integers m and l. We now consider just one horizontal spectra component, and suppress the term (5.2.28) which multiplies all perturbation variables. When initial perturbation is expanded, we get the sum of normal mode-like structures:

$$\Pi'(z,N) = \sum_{n\geq 1} \Pi_n(N)\left(\Gamma \sin(n'z) - n'\cos(n'z)\right)e^{Tz} + \Pi_0(N) + \Psi(z) \tag{5.2.23}$$

$$u'(z,N) = \sum_{n\geq 1} u_n\left(\Gamma \sin(n'z) - n'\cos(n'z)\right)e^{\frac{1}{2}az} + u_0(aN)e^{kaz} - \frac{C_p\theta_s}{f} il'\Psi(z) \tag{5.2.24}$$

$$v'(z,N) = \sum_{n\geq 1} v_n(N)\left(\Gamma \sin(n'z) - n'\cos(n'z)\right)e^{\frac{1}{2}az} + v_0(N)e^{kaz} + \frac{C_p\theta_s}{f} im'\Psi(z) \tag{5.2.25}$$

$$w'(z,N) = \sum_{n\geq 1} w_n(N)\sin(n'z)e^{\frac{1}{2}az} \tag{5.2.26}$$

$$\theta'(z,N) = \sum_{n\geq 1} \theta_n(N)\sin(n'z)e^{\left(\frac{1}{2}+k\right)az} + \frac{\theta_s}{ka\Pi_s}\frac{\partial\Psi}{\partial z} \tag{5.2.27}$$

$N = 0$, and $n' = \frac{\pi n}{H}$ for integer n. $\Gamma = \left(\frac{1}{2} - k\right)a$ and the stationary state Ψ satisfies

$$\theta'(z,0) = \frac{\theta_s}{ka\Pi_s}\frac{\partial\Psi}{\partial z} \tag{5.2.28}$$

At $z = 0, z = H$.

Substituting (5.2.29)-(5.2.33) into equations (5.2.20)-(5.2.24) using equations (5.2.25)-(5.2.26) with $F_d = F$ and using the fact that $\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial y}$ becomes multiplication by im' and il' respectively, for $n \geq 1$, we obtain the internal modes;

$$\Pi_n(N+1) - \Pi_n(N) = \frac{k}{1-k}\left(\overline{w}_n^{\alpha_2} - im'\overline{u}_n^{\alpha_1} - il'\overline{v}_n^{\alpha_1}\right)\Delta t \tag{5.2.29}$$

$$u_n(N+1) - u_n(N) = \left(-im'C_pT_s\overline{\Pi}_n^{\alpha_3} + f\overline{u}_n^{\alpha_3}\right)\Delta t \tag{5.2.30}$$

$$v_n(N+1) - v_n(N) = \left(-il'C_pT_s\overline{\Pi}_n^{\alpha_3} - f\overline{u}_n^{\alpha_3}\right)\Delta t \tag{5.2.31}$$

$$w_n(N+1) - w_n(N) = \left(-C_pT_s|q|^2\overline{\Pi}_n^{\alpha_4} + C_pka\overline{\theta}_n^{\alpha_4}\right)\Delta t \tag{5.2.32}$$

$$\theta_n(N+1) - \theta_n(N) = -T_ska\overline{w}_n^{\alpha_2}\Delta t \tag{5.2.33}$$

$q = \frac{a}{2} - ka + in'$. For external mode, that is, $n = 0$ we obtain the time-step relations:

$$\prod_0 (N+1) - \prod_0 (N) = -\frac{k}{1-k} i \left(m' \bar{u}_0^{\alpha_1} + l' \bar{v}_0^{\alpha_1} \right) \Delta t \tag{5.2.34}$$

$$u_0 (N+1) - u_0 (N) = \left(-im' C_p T_s \bar{\bar{\prod}}_0^{\alpha_3} + f \bar{v}_0^{\alpha_3} \right) \Delta t \tag{5.2.35}$$

$$v_0 (N+1) - v_0 (N) = \left(-il' C_p T_s \bar{\bar{\prod}}_0^{\alpha_3} - f \bar{u}_0^{\alpha_3} \right) \Delta t \tag{5.2.36}$$

To obtain the propagator of the SISL scheme, we set:

$$\mathbf{X}(N) = \begin{bmatrix} \prod_n (N) \\ i u_n (N) \\ i v_n (N) \\ w_n (N) \\ \theta_n (N) \end{bmatrix} \tag{5.2.37}$$

$$\mathbf{k} = \begin{bmatrix} 0 & \frac{m'k}{1-k} & \frac{l'k}{1-k} & \frac{-k}{1-k} & 0 \\ -C_p T_s m' & 0 & -f & 0 & 0 \\ -C_p T_s l' & f & 0 & 0 & 0 \\ C_p T_s |q|^2 & 0 & 0 & 0 & -C_p ka \\ 0 & 0 & 0 & T_s ka & 0 \end{bmatrix} \Delta t \tag{5.2.38}$$

And

$$\mathbf{A}_5 = \begin{bmatrix} 0 & \alpha_1 & \alpha_1 & \alpha_2 & 0 \\ \alpha_3 & 0 & \alpha_3 & 0 & 0 \\ \alpha_3 & \alpha_3 & 0 & 0 & 0 \\ \alpha_4 & 0 & 0 & 0 & \alpha_4 \\ 0 & 0 & 0 & \alpha_2 & 0 \end{bmatrix} \tag{5.2.39}$$

Let $\mathbf{L}_5^{\alpha} = \mathbf{A}_5 \circ \mathbf{L}_5$. In this case, $\circ$ is the Hadamard product. Setting $\mathbf{K}^{\alpha} = \mathbf{A}_5 \circ \mathbf{K}$, the relations (5.2.35)-(5.2.39) may be written:

$$\mathbf{X}(N+1) - \mathbf{X}(N) = -\mathbf{K}\mathbf{X}(N) - \mathbf{K}^{\alpha} \left(\mathbf{X}(N+1) - \mathbf{X}(N) \right) \tag{5.2.40}$$

Therefore,

$$\mathbf{X}(N+1) = \left(\mathbf{I} - \left(\mathbf{I} + \mathbf{K}^{\alpha} \right)^{-1} \mathbf{K} \right) \mathbf{X}(N) \tag{5.2.41}$$

Finally, we let E be a scalar multiple of

$$\begin{bmatrix} \frac{c_s}{Yk} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{|q|} & 0 \\ 0 & 0 & 0 & 0 & \frac{c_s}{|q|T_s(Yk)^{\frac{1}{2}}} \end{bmatrix} \tag{5.2.42}$$

Here, $Y = \frac{1}{(1-k)}$ and $c_s^2 = \gamma R T_s$. This considerations lead to the following expression

$$E\left(\mathbf{I} - \left(\mathbf{I} + \mathbf{K}^{\alpha}\right)^{-1}\mathbf{K}\right)E^{-1} = \mathbf{I} - \left(\mathbf{I} + \mathbf{L}^{\alpha}\right)^{-1}\mathbf{L} \tag{5.2.43}$$

Where

$$\mathbf{L} \equiv E\mathbf{K}E^{-1} = \begin{bmatrix} 0 & m'c_s & l'c_s & -|q|c_s & 0 \\ m'c_s & 0 & -f & 0 & 0 \\ -l'c_s & f & 0 & 0 & 0 \\ |q|c_s & 0 & 0 & 0 & -N_s \\ 0 & 0 & 0 & N_s & 0 \end{bmatrix} \Delta t \tag{5.2.44}$$

$N_s^2 = \frac{g^2}{C_p T_s}$ and

$$\mathbf{L}^{\alpha} \equiv E\mathbf{K}^{\alpha}E^{-1} = \mathbf{A}_5 \circ \mathbf{L} \tag{5.2.45}$$

In this case, E amounts to energy weighting.

We now investigate the external mode. Set

$$\mathbf{X}(N) = \begin{bmatrix} \Pi_0(N) \\ iu_0(N) \\ iv_0(N) \end{bmatrix} \tag{5.2.46}$$

This satisfies an equation of the form (5.2.47), where

$$\mathbf{K}=\begin{bmatrix} 0 & \dfrac{m'k}{1-k} & \dfrac{l'k}{1-k} \\ -C_pT_sm' & 0 & -f \\ -C_pT_sl' & f & 0 \end{bmatrix}\Delta t \tag{5.2.47}$$

And $\mathbf{K}^{\alpha}=\mathbf{A}_3\circ\mathbf{K}$, where $\mathbf{A}_3$ is the top-left sub-matrix of $\mathbf{A}_5$. If we choose E to be a scalar multiple of

$$\begin{bmatrix} \dfrac{c_s}{\gamma k} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.2.48}$$

An equation similar to (5.2.48) is satisfied, and in this case;

$$\mathbf{L}=\begin{bmatrix} 0 & m'c_s & l'c_s \\ -m'c_s & 0 & -f \\ -l'c_s & f & 0 \end{bmatrix}\Delta t \tag{5.2.49}$$

And $\mathbf{L}^{\alpha}=\mathbf{A}_3\circ\mathbf{L}$.

L and L^{α} in (5.2.50) and (5.2.51) have the form (5.2.10), and so are $\mathbf{L}$ and $\mathbf{L}^{\alpha}$ in (5.2.55). Hence if condition

$$\begin{aligned} &\alpha_3\geq\alpha_1\geq\frac{1}{2} \\ &\alpha_2,\alpha_4\geq\frac{1}{2} \end{aligned} \tag{5.2.50}$$

Holds, then by **theorem 2** all internal and external modes of the discretisation are stable.

Furthermore, let $f=0$ and $\dfrac{\partial}{\partial y}=0$. Using a similar process as above, we arrive at the equation of internal modes

$$\mathbf{X}(N)=\begin{bmatrix} \Pi_n(N) \\ iu_n(N) \\ w_n(N) \\ \theta_n(N) \end{bmatrix} \tag{5.2.51}$$

The resulting equation is of the same form as (5.2.47). $\mathbf{K}$ and $\mathbf{K}^{\alpha}$ are related to $\mathbf{L}$ and $\mathbf{L}^{\alpha}$ as in equation (5.2.48), where now $\mathbf{L}\equiv E\mathbf{K}E^{-1}$ and $\mathbf{L}^{\alpha}\equiv E\mathbf{K}^{\alpha}E^{-1}$ are 4 by 4 matrices formed

by deleting the middle row and column of (5.2.50) and (5.2.51) respectively. $\mathbf{L}$ and $\mathbf{L}^{\alpha}$ have the form (5.2.8). As $f = 0$ and $l' = 0$, the matrices $\mathbf{L}$ and $\mathbf{L}^{\alpha}$ for the external mode given in equation (5.2.55) have the form (5.2.8) with $b = c = 0$. Hence, if the condition $\alpha_1, \alpha_2, \alpha_3, \alpha_4 \geq \frac{1}{2}$ holds, then by **Theorem 1 ,** all internal and external modes of discretisation with $f = 0$ and $\frac{\partial}{\partial y} = 0$ are stable.

5.3. Convergence of the Semi-Implicit Semi-Lagrangian scheme

The analysis of convergence of the discretised equation using the SISL scheme is hard given the number of terms and the nature of these equations. However, to circumvent this problem, we are going to use Lax Equivalence Theorem by Peter Lax as a way of escape. We shall begin by stating this theorem without proof.

5.3.1. Theorem 3: Lax Equivalence Theorem

It is also called Lax-Richtmyer Theorem. It states that for a consistent finite difference method for a well-posed linear initial value problem, the method is convergent if and only if it is stable.

We note that to the extent to which the scheme is linear, consistent and stable, to that extent, it is convergent. Therefore, having shown the consistency and stability of this scheme, we conclude that the linear part of the discretised scheme is convergent.

CHAPTER 6: THE HELMHOLTZ EQUATION

As it has been pointed out in chapter 4, the discretisation of governing equations leads to the Helmholtz equation, which is solved for the increment in pressure field. Solving of this equation alone takes 35% of the forecast time. As a result, optimal and efficient methods are being sought for solving this equation. In this chapter, we are going to solve a simple sample example of the Helmholtz problem using Gauss-Seidel, Jacobi, Successive-Over-Relaxation and Conjugate Gradient Methods. Comparison of these methods will be done in regards to efficiency. Before then, let us analyse the stability of the 2-dimensional Helmholtz equation.

6.1. Analysis of stability of the Helmholtz equation

We seek to analyse the stability of the simplified form of Helmholtz equation (4.4.1).

$$\nabla^2 \Pi'(x,y) + \omega^2 \Pi'(x,y) = \Phi(x,y) \tag{6.1.1}$$

$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial x^2}$ is the Laplacian in rectangular coordinates, ω is the wave number, Π' is the increment in pressure field and Φ represent the value of the pressure field calculated from the previous time step.

Let $\Phi(x,y) = 0$. Therefore,

$$\nabla^2 \Pi'(x,y) + \omega^2 \Pi'(x,y) = 0 \tag{6.1.2}$$

This simplified form of equation (6.1.1) eases the analysis of stability. Central difference approximation of (6.1.2) yields the following difference equation:

$$\frac{\Pi'_{i+1,j} - 2\Pi'_{i,j} + \Pi'_{i-1,j}}{(\Delta x)^2} + \frac{\Pi'_{i,j+1} - 2\Pi'_{i,j} + \Pi'_{i,j-1}}{(\Delta y)^2} + \omega^2 \Pi'_{i,j} = 0 \tag{6.1.3}$$

Let $\Delta x = \Delta y = h$. This substitution and multiplication by h^2 leads to:

$$\Pi'_{i+1,j} - 2\Pi'_{i,j} + \Pi'_{i-1,j} + \Pi'_{i,j+1} - 2\Pi'_{i,j} + \Pi'_{i,j-1} + (h\omega)^2 \Pi'_{i,j} = 0 \tag{6.1.4}$$

Collecting like terms together in (6.1.4) leads to:

$$(4 - h^2\omega^2)\Pi'_{i,j} = \Pi'_{i+1,j} + \Pi'_{i-1,j} + \Pi'_{i,j+1} + \Pi'_{i,j-1} \tag{6.1.5}$$

Suppose we use Jacobi iteration for (6.1.5). The recurrence relation for the Helmholtz equation (6.1.2) is then given by:

$$\Pi'^{(k+1)}_{i,j} = \frac{1}{(4 - h^2\omega^2)}\left[\Pi'^{(k)}_{i+1,j} + \Pi'^{(k)}_{i-1,j} + \Pi'^{(k)}_{i,j+1} + \Pi'^{(k)}_{i,j-1}\right] \tag{6.1.6}$$

For stability analysis, we use Von Neumann's method.

Let us define an error at the k^{th} iteration as

$$e_{i,j}^{(k)} = \Pi_{i,j}^{\prime(k)}\left(\text{Exact}\right) - \Pi_{i,j}^{\prime(k)}\left(\text{Approximated}\right) \tag{6.1.7}$$

Since the exact solution satisfies the relation (6.1.6), so is the error. Hence,

$$e_{i,j}^{(k+1)} = \frac{1}{\left(4-h^2\omega^2\right)}\left[e_{i+1,j}^{(k)} + e_{i-1,j}^{(k)} + e_{i,j+1}^{(k)} + e_{i,j-1}^{(k)}\right] \tag{6.1.8}$$

Let the error equation have a solution of the form

$$e_{l,m}^{(k)} = A_{pq}\sin\frac{p\pi l}{m}\sin\frac{q\pi m}{m},\ 1\le p,q\le m-1 \tag{6.1.9}$$

A_{pq} is an arbitrary constant.

$$\begin{aligned} e_{i+1,j}^{(k)} &= A_{pq}\sin\frac{p\pi\left(i+1\right)}{m}\sin\frac{q\pi j}{m} \\ &= A_{pq}\sin\frac{q\pi j}{m}\left[\sin\frac{p\pi i}{m}\cos\frac{p\pi}{m}+\cos\frac{p\pi i}{m}\sin\frac{p\pi}{m}\right] \end{aligned} \tag{6.1.10}$$

$$\begin{aligned} e_{i-1,j}^{(k)} &= A_{pq}\sin\frac{p\pi\left(i-1\right)}{m}\sin\frac{q\pi j}{m} \\ &= A_{pq}\sin\frac{q\pi j}{m}\left[\sin\frac{p\pi i}{m}\cos\frac{p\pi}{m}-\cos\frac{p\pi i}{m}\sin\frac{p\pi}{m}\right] \end{aligned} \tag{6.1.11}$$

$$\begin{aligned} e_{i+1,j}^{(k)} + e_{i-1,j}^{(k)} &= 2A_{pq}\sin\frac{q\pi i}{m}\sin\frac{p\pi i}{m}\cos\frac{p\pi}{m} \\ &= 2\cos\frac{p\pi}{m}\left[A_{pq}\sin\frac{p\pi i}{m}\sin\frac{q\pi j}{m}\right] = 2\cos\frac{p\pi}{m}\mathrm{e}_{i,j}^{(k)} \end{aligned} \tag{6.1.12}$$

In the selfsame way,

$$e_{i,j+1}^{(k)} + e_{i,j-1}^{(k)} = 2A_{pq}\cos\frac{q\pi}{m}\sin\frac{p\pi i}{m}\sin\frac{q\pi j}{m} = 2\cos\frac{q\pi}{m}e_{i,j}^{(k)} \tag{6.1.13}$$

Substituting equations (6.1.12) and (6.1.13) into equation (6.1.8) yields:

$$\begin{aligned} e_{i,j}^{(k+1)} &= \frac{1}{\left(4-\omega^2h^2\right)}\left[2\cos\frac{p\pi}{m}+2\cos\frac{q\pi}{m}\right]e_{i,j}^{(k)} \\ &= \xi e_{i,j}^{(k)} \text{ where } \xi = \frac{2}{\left(4-\omega^2h^2\right)}\left[\cos\frac{p\pi}{m}+\cos\frac{q\pi}{m}\right] \end{aligned} \tag{6.1.14}$$

ξ is known as the propagating factor.

For stability;

$$\left|\frac{e_{i,j}^{(k+1)}}{e_{i,j}^{(k)}}\right| = |\xi| < 1 \tag{6.1.15}$$

This implies that

$$\left|\frac{2}{\left(4-\omega^2 h^2\right)}\left[\cos\frac{p\pi}{m}+\cos\frac{q\pi}{m}\right]\right| < 1 \tag{6.1.16}$$

Let $p = q = 1$. Therefore,

$$-1 < \frac{4}{4-\omega^2 h^2}\cos\frac{\pi}{m} < 1 \tag{6.1.17}$$

(6.1.17) is the condition for stability of the Helmholtz equation (6.1.2). It shows that the wave number ω and the spatial step-size h determine the stability of the numerical scheme. To guarantee stability, the wave number and the spatial step-sizes should be chosen to satisfy (6.1.17).

6.2. Solving the Helmholtz equation

In this part, we seek a numerical solution of equation (6.1.1);

$$\nabla^2 \prod'(x,y) + \omega^2 \prod'(x,y) = \Phi(x,y) \tag{6.2.1}$$

With boundary conditions;

$$\prod'(x,y) = \ln\left[(x+1)^2 + y^2\right] \tag{6.2.2}$$

Let $\Phi(x,y) = \frac{1}{4}(x+y)$, $\Delta x = \Delta y = 0.25$ and $\omega = 1$. Furthermore, let this problem be solved on a unit square $0 \le x, y \le 1$.

This problem is a simple form of that which is used in Numerical Weather Prediction. We have deliberately taken a simple case for the purpose of our analysis, to give a 'feel' of what happens in large scale.

Central difference discretisation of this problem and rearranging in the form

$$\mathbf{A}\prod' = \mathbf{b} \tag{6.2.3}$$

Leads to the expression;

$$\begin{bmatrix} 3.9375 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3.9375 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 3.9375 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 3.9375 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 3.9375 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 3.9375 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 3.9375 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 3.9375 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 3.9375 \end{bmatrix} \begin{bmatrix} \prod_1' \\ \prod_2' \\ \prod_3' \\ \prod_4' \\ \prod_5' \\ \prod_6' \\ \prod_7' \\ \prod_8' \\ \prod_9' \end{bmatrix} = \begin{bmatrix} 0.6319117244 \\ 0.9984302162 \\ 2.771030124 \\ 0.4106435513 \\ 0.25 \\ 1.759418983 \\ 1.637270447 \\ 1.491154996 \\ 3.294669267 \end{bmatrix} \quad (6.2.4)$$

In the following parts, the system of equations (6.2.4) is solved using Jacobi Iterative, Gauss-Seidel, Successive-Over-Relaxation and Conjugate Gradient methods.

6.2.1. Using Jacobi Iterative (JI) Method

Given the system of equations (6.2.3), **A** can be decomposed into the diagonal matrix, **D,** the upper triangular matrix, **U** and the lower triangular matrix, **L,** that is,

$$\mathbf{Ax} = (\mathbf{L} + \mathbf{D} + \mathbf{U})\mathbf{x} = \mathbf{b} \quad (6.2.5)$$

Simple manipulation of (6.2.5) yields;

$$\mathbf{x} = \mathbf{D}^{-1}\left[\mathbf{b} - (\mathbf{L} + \mathbf{U})\mathbf{x}\right] \quad (6.2.6)$$

(6.2.6) can be written iteratively as follows:

$$\mathbf{x}^{(k+1)} = \mathbf{D}^{-1}\left[\mathbf{b} - (\mathbf{L} + \mathbf{U})\mathbf{x}^{(k)}\right], \quad k = 0,1,2,... \quad (6.2.7)$$

Most precisely, (6.2.7) can be cast in the form:

$$\mathbf{x}^{(k+1)} = \mathbf{c} + \mathbf{T}_j\mathbf{x}^{(k)}, \; k = 1,2,... \quad (6.2.8)$$

In this case, $\mathbf{c} = \mathbf{D}^{-1}\mathbf{b}$ and the Jacobi iteration matrix $\mathbf{T}_j = -\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})$.

Using (6.2.8), equation (6.2.4) can be solved in MATLAB. The solution is given in the table below. The MATLAB codes used are in Appendix A.

Table 1: MATLAB output for JI Method solution of Helmholtz equation

k	$\prod_1'^{(k)}$	$\prod_2'^{(k)}$	$\prod_3'^{(k)}$	$\prod_4'^{(k)}$	$\prod_5'^{(k)}$	$\prod_6'^{(k)}$	$\prod_7'^{(k)}$	$\prod_8'^{(k)}$	$\prod_9'^{(k)}$
1	0.1605	0.2536	0.7038	0.1043	0.0635	0.4468	0.4158	0.3787	0.8367
2	0.2514	0.4892	0.8816	0.2668	0.3640	0.8542	0.5385	0.7129	1.0464
3	0.3525	0.6338	1.0449	0.3973	0.6535	1.0290	0.6646	0.8737	1.2347
4	0.4224	0.7744	1.1260	0.5286	0.8086	1.1918	0.7386	1.0271	1.3199

5	0.4914	0.8522	1.2031	0.6045	0.9579	1.2734	0.8109	1.1069	1.4003
6	0.5304	0.9272	1.2436	0.6783	1.0379	1.3513	0.8504	1.1835	1.4413
7	0.5682	0.9677	1.2824	0.7186	1.1150	1.3923	0.8887	1.2243	1.4805
8	0.5888	1.0068	1.3031	0.7575	1.1563	1.4317	0.9093	1.2636	1.5013
9	0.6085	1.0277	1.3230	0.7784	1.1961	1.4527	0.9291	1.2846	1.5213
.	.								
.	.								
.	.								
34	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645
35	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645

6.2.2. Using Gauss-Seidel Iterative (GSI) Method

Generally, for n linear equations, the Gauss-Seidel Iterative Method is defined as:

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{k} \right] \tag{6.2.9}$$

$$i = 1, 2, ..., n;\ k = 0, 1, 2, ...$$

In matrix form, the Gauss-Seidel Iterative Method is given by:

$$\mathbf{x}^{(k+1)} = \mathbf{T}_G \mathbf{x}^{(k)} + \mathbf{c} \tag{6.2.10}$$

where $\mathbf{T}_G = -(\mathbf{D}+\mathbf{L})^{-1}\mathbf{U}$ and $\mathbf{c} = (\mathbf{D}+\mathbf{L})^{-1}\mathbf{b}$

This method is also known as the method of successive iteration because the most recent values of all x_i are used in the calculation.

The solution of (6.2.4) using Gauss-Seidel Iterative Method given by (6.2.10) in MATLAB is given below. The MATLAB codes used in this part are in Appendix A.

Table 2: MATLAB output for GSI Method solution of Helmholtz equation

k	$\Pi_1'^{(k)}$	$\Pi_2'^{(k)}$	$\Pi_3'^{(k)}$	$\Pi_4'^{(k)}$	$\Pi_5'^{(k)}$	$\Pi_6'^{(k)}$	$\Pi_7'^{(k)}$	$\Pi_8'^{(k)}$	$\Pi_9'^{(k)}$
1	0.1605	0.2943	0.7785	0.1450	0.1751	0.6890	0.4527	0.5381	1.1484
2	0.2721	0.5648	1.0222	0.3328	0.6031	1.1513	0.6370	0.9853	1.3794
3	0.3885	0.7650	1.1904	0.5179	0.9319	1.3362	0.7976	1.1683	1.4728
4	0.4863	0.9161	1.2758	0.6670	1.1016	1.4247	0.8819	1.2565	1.5177
5	0.5625	1.0002	1.3196	0.7509	1.1891	1.4694	0.9256	1.3012	1.5404
6	0.6052	1.0444	1.3422	0.7951	1.2343	1.4924	0.9482	1.3242	1.5521
7	0.6277	1.0673	1.3538	0.8180	1.2576	1.5042	0.9599	1.3361	1.5581
8	0.6393	1.0792	1.3599	0.8298	1.2697	1.5104	0.9659	1.3422	1.5612
.	.								
.	.								

.	.								
17	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645
18	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645

6.2.3. Using Successive-Over-Relaxation (SOR) Method

This method is one of the techniques for improving the convergence of Gauss-Seidel Method. It is generally given by:

$$x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \frac{\omega}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right] \tag{6.2.11}$$

$$i = 1, 2, ..., n;\ k = 1, 2, ...$$

In matrix form, (6.2.11) can be written as follows;

$$\mathbf{x}^{(k+1)} = \mathbf{T}_{\omega}\mathbf{x}^{(k)} + \mathbf{c} \tag{6.2.12}$$

where

$$\mathbf{T}_{\omega} = (\mathbf{D} + \omega\mathbf{L})^{-1}\left[(1-\omega)\mathbf{D} - \omega\mathbf{U}\right] \text{ and } \mathbf{c} = \omega(\mathbf{D} - \omega\mathbf{L})^{-1}\mathbf{b}$$

ω is called the relaxation factor. It can be shown that convergence is obtained within the range $0 < \omega < 2$. To accelerate convergence of an already convergent Gauss-Seidel Method, over-relaxation method is used, that is, by choosing ω in the interval $1 < \omega < 2$.

The solution of (6.2.4) using Successive-Over-Relaxation equation (6.2.12) in MATLAB is in table 3 below. The MATLAB codes used are in Appendix A.

Table 3: MATLAB output for SOR Method solution of the Helmholtz equation

k	$\prod_1'^{(k)}$	$\prod_2'^{(k)}$	$\prod_3'^{(k)}$	$\prod_4'^{(k)}$	$\prod_5'^{(k)}$	$\prod_6'^{(k)}$	$\prod_7'^{(k)}$	$\prod_8'^{(k)}$	$\prod_9'^{(k)}$
1	0.2054	0.3913	1.0280	0.2003	0.2736	0.9951	0.5973	0.7679	1.6441
2	0.3402	0.7487	1.1798	0.4711	0.9743	1.5281	0.7678	1.3705	1.5529
3	0.5067	0.9799	1.3857	0.7326	1.3074	1.5244	1.0010	1.3562	1.5726
4	0.6202	1.1273	1.3748	0.8804	1.3043	1.5273	0.9791	1.3585	1.5688
5	0.6844	1.1023	1.3707	0.8517	1.2894	1.5190	0.9766	1.3510	1.5648
6	0.6490	1.0916	1.3657	0.8426	1.2820	1.5160	0.9719	1.3478	1.5639
7	0.6525	1.0917	1.3661	0.8424	1.2820	1.5167	0.9721	1.3485	1.5646
8	0.6515	1.0915	1.3662	0.8422	1.2823	1.5169	0.9722	1.3487	1.5645
.	.								
.	.								
.	.								
12	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645
13	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645

6.2.4. Using Conjugate Gradient (CG) Method

Let **A** be positive definite and orthogonal with respect to inner product notation

$$< \mathbf{x}, \mathbf{y} >= \mathbf{x}^T \mathbf{A}\mathbf{y} \tag{6.2.13}$$

where $\mathbf{x}$ and $\mathbf{y}$ are $n-$dimensional vectors

Let the following hold for **x** and **y.**

$$< \mathbf{x}, \mathbf{A}\mathbf{y} > \; = \; < \mathbf{A}\mathbf{x}, \mathbf{y} > \tag{6.2.14}$$

We seek the vector $\mathbf{x}^*$, a solution to the system (6.2.4) if and only if it minimises the error

$$E(x) = \; < \mathbf{x}, \mathbf{A}\mathbf{x} > -2 < \mathbf{x}, \mathbf{b} > \tag{6.2.15}$$

Furthermore, the function $E(\mathbf{x}+t\mathbf{v})$ has a minimum for any $\mathbf{x}, \mathbf{v} \neq \mathbf{0}$ when

$$t = \frac{< \mathbf{v}, \mathbf{b} - \mathbf{A}\mathbf{x} >}{< \mathbf{v}, \mathbf{A}\mathbf{v} >} \tag{6.2.16}$$

Starting this process involves specifying an initial estimate and computing the initial residue vector as follows:

$$\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)} \tag{6.2.17}$$

Improved estimates are obtained iteratively as follows:

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + t_k \mathbf{v}^{(k)} \tag{6.2.18}$$

$\mathbf{v}^{(k)}$ is a search direction expressed in vector form.

$$t_k = \frac{< \mathbf{v}^{(k)}, \mathbf{b} - \mathbf{A}\mathbf{x}^{(k-1)} >}{< \mathbf{v}^{(k)}, \mathbf{A}\mathbf{v}^{(k)} >} \tag{6.2.19}$$

is chosen so that $E(\mathbf{x}^{(k)})$ is minimised.

We use $\mathbf{v}^{(1)} = \mathbf{r}^{(0)}$ only at the beginning of the process. For subsequent iterations, we use the following expression.

$$\mathbf{v}^{(k+1)} = \mathbf{r}^{(k)} + \frac{\left\| \mathbf{r}^{(k)} \right\|^2}{\left\| \mathbf{r}^{(k-1)} \right\|^2} \mathbf{v}^{(k)} \tag{6.2.20}$$

Therefore, by using the Conjugate Gradient Method to solve (6.2.4) in MATLAB, we obtain the following.

Table 4: MATLAB output for CG Method solution of the Helmholtz equation

k	$\Pi_1'^{(k)}$	$\Pi_2'^{(k)}$	$\Pi_3'^{(k)}$	$\Pi_4'^{(k)}$	$\Pi_5'^{(k)}$	$\Pi_6'^{(k)}$	$\Pi_7'^{(k)}$	$\Pi_8'^{(k)}$	$\Pi_9'^{(k)}$
1	0.2787	0.4404	1.2223	0.1811	0.1103	0.7761	0.7222	0.6578	1.4533
2	0.4337	0.9360	1.3004	0.5637	0.8928	1.6281	0.8162	1.3499	1.5411
3	0.5560	1.0073	1.3623	0.7047	1.3007	1.5669	0.9085	1.3388	1.6024
4	0.6211	1.0937	1.3661	0.8467	1.2863	1.5141	0.9536	1.3465	1.5789
5	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645
6	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645

6.2.5. Using Bi-Conjugate Gradient Stabilised (BiCGSTAB) Method

This method avoids the often irregular convergence patterns of the Conjugate Gradient Squared method. It computes $i \to \mathbf{Q}_i(\mathbf{A})\mathbf{P}_i(\mathbf{A})\mathbf{r}^{(0)}$ where $\mathbf{Q}_i$ is an ith degree polynomial describing a deepest descent.

The following is a MATLAB output for (6.2.4) using the Bi-Conjugate Gradient Stabilized method.

Table 5: MATLAB output for Helmholtz equation using BiCGSTAB Method

k	$\Pi_1'^{(k)}$	$\Pi_2'^{(k)}$	$\Pi_3'^{(k)}$	$\Pi_4'^{(k)}$	$\Pi_5'^{(k)}$	$\Pi_6'^{(k)}$	$\Pi_7'^{(k)}$	$\Pi_8'^{(k)}$	$\Pi_9'^{(k)}$
1	0.6514	1.0914	1.3658	0.8420	1.2820	1.5164	0.9719	1.3482	1.5640
2	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645
3	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645

6.2.6. Bi-Conjugate Gradient Method (BICG)

The residuals in Conjugate Gradient method are replaced with relations that are similar but based on $\mathbf{A}^T$ instead of $\mathbf{A}$. Two sequences of residuals are updated.

$$\mathbf{r}^{(i)} = \mathbf{r}^{(i-1)} - \alpha_i \mathbf{A}\mathbf{p}^{(i)},\ \tilde{\mathbf{r}}^{(i)} = \tilde{\mathbf{r}}^{(i-1)} - \alpha_i \mathbf{A}^T \tilde{\mathbf{p}}^{(i)} \tag{6.2.21}$$

The sequences of search directions are

$$\mathbf{p}^{(i)} = \mathbf{r}^{(i-1)} + \beta_{i-1}\mathbf{p}^{(i-1)},\ \tilde{\mathbf{p}}^{(i)} = \tilde{\mathbf{r}}^{(i-1)} + \beta_{i-1}\tilde{\mathbf{p}}^{(i-1)} \tag{6.2.22}$$

where

$$\alpha_i = \frac{\tilde{\mathbf{r}}^{(i-1)^T}\mathbf{r}^{(i-1)}}{\tilde{\mathbf{p}}^{(i)^T}\mathbf{A}\mathbf{p}^{(i)}},\ \beta_i = \frac{\tilde{\mathbf{r}}^{(i)^T}\mathbf{r}^{(i)}}{\tilde{\mathbf{r}}^{(i-1)^T}\mathbf{r}^{(i-1)}}$$

which ensure the bi-orhogonality relations

$$\tilde{\mathbf{r}}^{(i)^T}\mathbf{r}^{(j)} = \tilde{\mathbf{p}}^{(i)^T}\mathbf{A}\mathbf{p}^{(j)} = 0 \quad \text{if } i \neq j$$

Using the Bi-Conjugate Gradient Method to solve (6.2.4) in MATLAB we obtain the following output.

Table 6: MATLAB output for the solution of the Helmholtz equation using BICG Method

k	$\Pi_1'^{(k)}$	$\Pi_2'^{(k)}$	$\Pi_3'^{(k)}$	$\Pi_4'^{(k)}$	$\Pi_5'^{(k)}$	$\Pi_6'^{(k)}$	$\Pi_7'^{(k)}$	$\Pi_8'^{(k)}$	$\Pi_9'^{(k)}$
1	0.3938	0.7570	1.3256	0.8626	0.6988	1.0954	1.2532	1.3280	1.5384
2	0.3938	0.7570	1.3256	0.8626	0.6988	1.0954	1.2532	1.3280	1.5384
3	0.6846	1.0839	1.3987	0.8987	1.3052	1.5566	0.9959	1.2426	1.5321
4	0.6838	1.1022	1.3777	0.8555	1.2799	1.5289	0.9632	1.2996	1.5649
5	0.6616	1.0886	1.3669	0.8457	1.2733	1.5214	0.9684	1.3297	1.5715
6	0.6543	1.0885	1.3663	0.8431	1.2766	1.5197	0.9700	1.3416	1.5683
7	0.6517	1.0926	1.3669	0.8413	1.2838	1.5157	0.9729	1.3483	1.5643
8	0.6516	1.0918	1.3662	0.8424	1.2824	1.5168	0.9724	1.3485	1.5646
9	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645
10	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645

6.2.7. Quasi-Minimal Residual (QMR) Method

It is meant to overcome the irregularity of the convergence of BICG method. It solves the reduced tri-diagonal system in a least squares sense. It uses look-ahead techniques to avoid breakdowns that make it more robust than BICG.

Table 7: MATLAB output of the solution to the Helmholtz equation using QMR Method

k	$\Pi_1'^{(k)}$	$\Pi_2'^{(k)}$	$\Pi_3'^{(k)}$	$\Pi_4'^{(k)}$	$\Pi_5'^{(k)}$	$\Pi_6'^{(k)}$	$\Pi_7'^{(k)}$	$\Pi_8'^{(k)}$	$\Pi_9'^{(k)}$
1	0.3598	0.6915	1.2110	0.7880	0.6383	1.0007	1.1448	1.2132	1.4054
2	0.3598	0.6915	1.2110	0.7880	0.6383	1.0007	1.1448	1.2132	1.4054
3	0.6657	1.0612	1.3881	0.8925	1.2664	1.5241	1.0045	1.2412	1.5251
4	0.6812	1.0963	1.3792	0.8609	1.2779	1.5282	0.9692	1.2912	1.5592
5	0.6641	1.0896	1.3684	0.8476	1.2739	1.5222	0.9685	1.3249	1.5699
6	0.6558	1.0886	1.3666	0.8438	1.2762	1.5200	0.9697	1.3390	1.5686
7	0.6521	1.0922	1.3668	0.8415	1.2832	1.5161	0.9727	1.3475	1.5647
8	0.6516	1.0919	1.3662	0.8424	1.2825	1.5168	0.9724	1.3485	1.5646
9	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645
10	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645

6.2.8. Generalised Minimal Residual (GMRES) Method

The iterates of this method are given by;

$$\mathbf{x}^{(i)} = \mathbf{x}^{(0)} + y_1\mathbf{v}^{(1)} + \ldots + y_i\mathbf{v}^{(i)} \tag{6.2.23}$$

The coefficients y_k are chosen to minimise the residual norm $\left\|\mathbf{b} - \mathbf{A}\mathbf{x}^{(i)}\right\|$.

The solution to (6.2.4) using the Generalised Minimal Residual Method in MATLAB is shown below.

Table 8: MATLAB output for the solution of the Helmholtz equation using GMRES Method

k	$\prod_1'^{(k)}$	$\prod_2'^{(k)}$	$\prod_3'^{(k)}$	$\prod_4'^{(k)}$	$\prod_5'^{(k)}$	$\prod_6'^{(k)}$	$\prod_7'^{(k)}$	$\prod_8'^{(k)}$	$\prod_9'^{(k)}$
1	0.6441	1.0837	1.3798	0.8277	1.2601	1.5188	0.9745	1.3469	1.5834
2	0.6516	1.0918	1.3662	0.8424	1.2825	1.5170	0.9722	1.3487	1.5645
3	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645
4	0.6517	1.0918	1.3663	0.8424	1.2825	1.5169	0.9723	1.3487	1.5645

CHAPTER 7: DATA ANALYSIS AND RESULTS

In Chapter 4, an overview of numerical aspect of Weather Forecasting has been highlighted. It begins with the primitive equations of hydrodynamics. These are then cast in a form that is amenable to description of the atmospheric flow. This process involves incorporating the Coriolis and centrifugal forces, factoring in the aspect of the rotation of the earth and consideration of the change in the position vectors of two points due to curvature effects of the grid. The result is a system of equations (4.1.49), (4.1.51), (4.1.53), (4.1.54) and (4.2.11)-(4.2.13).

The resultant equations are discretised using the off-centred, Semi-Implicit, Semi-Lagrangian method. Unlike the Eulerian decomposition that sustains the advection term, the SISL scheme retains intact the material derivative. Moreover, the SISL scheme is stable even when long time steps are taken. However, it is subject to numerical instabilities under certain extrapolation procedures. Discretisation of governing equations involves series of predictor corrector steps, especially for terms that cannot be readily obtained.

Successful discretisation of governing equations yields $13N+7$ system of equations where N is the number of grid points considered. This system of equations contains 13 variables from which 8 are prognostically determined while 5 of them are obtained diagnostically. The system of equations is then substituted to the equation of state and the result is a Helmholtz problem. Due to the rigidity of the boundary, that is, the tropopause and the Earth's surface, the Neumann boundary conditions are inferred. The solution of the Helmholtz equation with the boundary conditions yields the increment in pressure field, $\prod'$. The pressure field increment is substituted back to the system of discretised equations to obtain the various prognostic variables.

Chapter 5 is concerned with the analysis of consistency, stability and convergence of SISL scheme. It was established that the scheme is consistent, stable given that $\alpha_1, \alpha_2, \alpha_3, \alpha_4 \geq \frac{1}{2}$ and is convergent by Lax Equivalence theorem.

In chapter 6, a simple 2-dimensional Helmholtz equation is chosen for analysis of stability and solution. It was established that for stability, the following condition holds:

$$-1 < \frac{4}{4-\omega^2 h^2}\cos\frac{\pi}{m} < 1 \qquad (7.1.1)$$

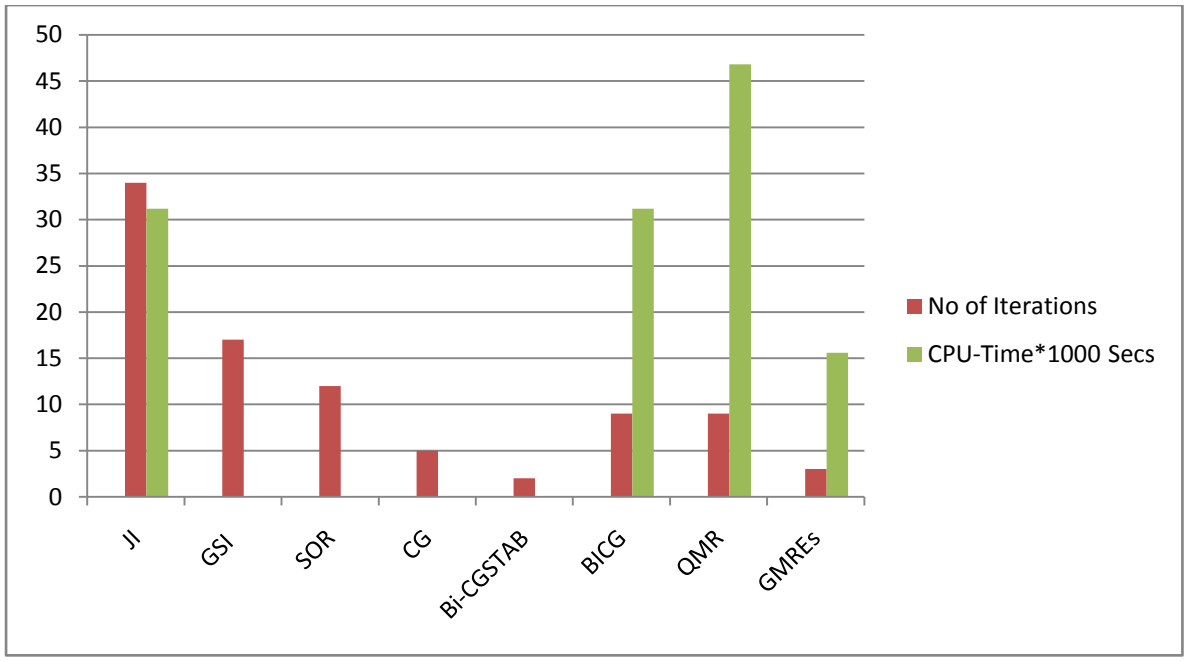
(7.1.1) shows that the stability of the Helmholtz equation depends on the spatial step-size, h, and the wave number, ω.

The solution of the sample 2-dimensional Helmholtz has been summarised in Table 5 and Graph 1 for the different methods used.

Table 9: A summary of the number of iterations for methods used

Method	JI	GSI	SOR	CG	Bi-CGSTAB	BICG	QMR	GMRES
No. of iterations	34	17	12	5	2	9	9	3
CPU-Time*1000 (Seconds)	31.2	0	0	0	0	31.2	46.8	15.6

Graph 1: A graph showing the number of iterations and CPU-time of different methods



The Helmholtz equation can be visualized using the Graphical User Interface in MATLAB. The following is an output of the Helmholtz equation with $\omega = 1$.

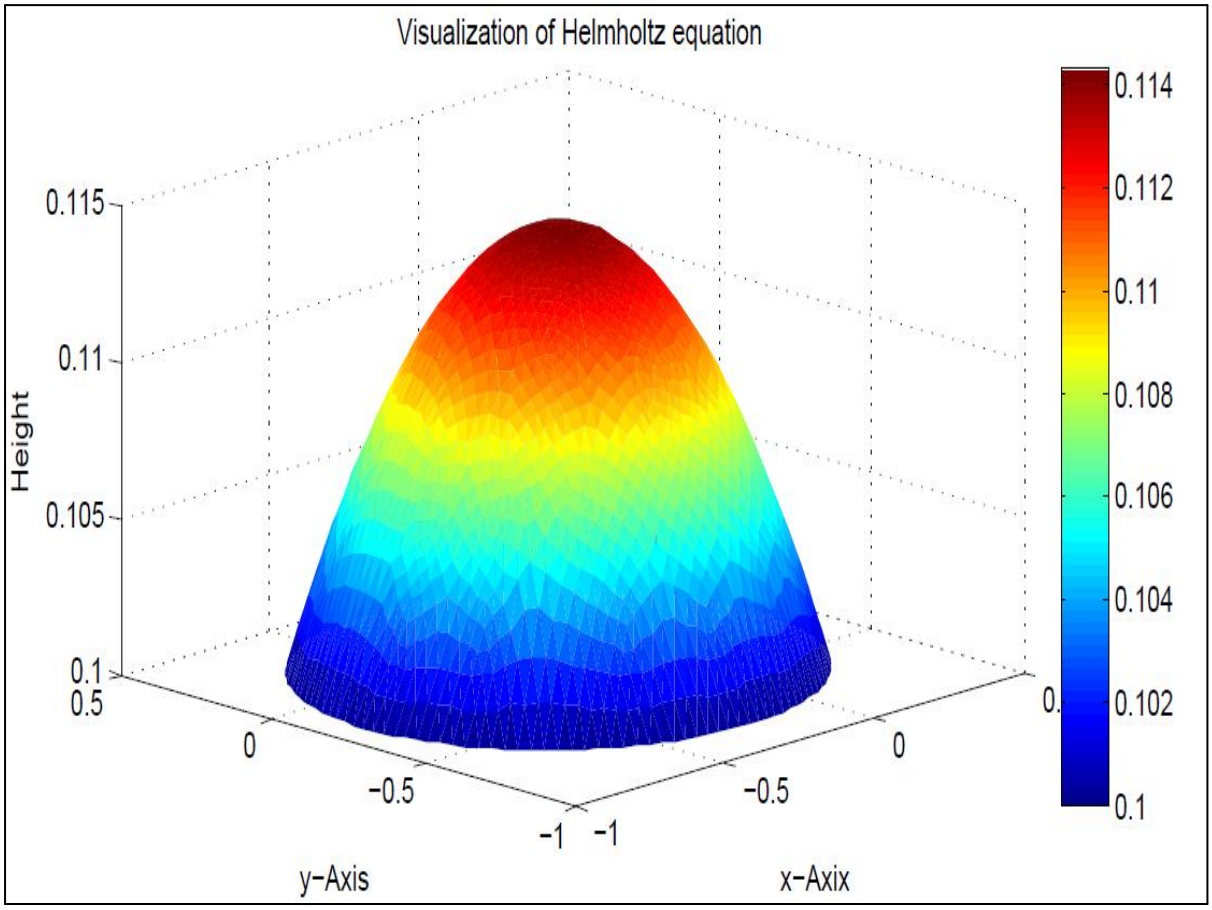


Figure 4: Visualization of the Helmholtz equation using Graphical User Interface in MATLAB

CHAPTER 8: FINDINGS AND RECOMMENDATIONS

8.2. Findings

From the study of Numerical Weather Prediction, particularly the Unified Model of UK Met Office, It was established that the Governing equations are discretised using the off-centred Semi-Implicit Semi-Lagrangian time discretisation method. Before discretisation of these equations, the primitive equations, which include the momentum equations, the continuity equation, the representation of moisture equation and the equation of state, are cast in a form amenable for practical implementation. The earth is estimated on a spherical surface. This is because the Newtonian gravitational force supersedes the Centrifugal force in lower atmosphere.

The SISL time discretisation is preferred for its stability even when long time-steps are taken and it doesn't have Eulerian advection terms. However, it may be subject to numerical instabilities if certain extrapolation procedures are used. A two-time-level scheme is used because it requires less storage, and for a given time step they reach a given forecast 50% fewer steps because successive intervals do not overlap. For any off-centred two-time-level scheme, the time weighting factor, α, is chosen such that, $\frac{1}{2} < \alpha \leq 1$. α is chosen closer to $\frac{1}{2}$ to render the scheme more accurate and less damping.

The source term in the SISL scheme is divided into the linear and non-linear terms. The linear part is dealt with by algebraic elimination while the non-linear part is accommodated using iterative procedures, which consist of a fixed number of predictor-corrector steps.

The number of discretised governing equations obtained is 13N+7, where N is the number of grid points used, with 13 variables. 8 of these variables are determined from the associated prognostic equation of the variable. These variables are u, v, w, ρ_y, θ, m_v, m_{cl} and m_{cf}. The other 5; $\dot{\eta}$, ρ, θ_v, $\prod$ and p are diagnostically determined. The coupling of these equations by algebraic decomposition yields a Helmholtz equation, which, when solved, yields the increment in pressure field, $\prod'$. By back substitution of the value of the increment in pressure field in the discretised equations, other prognostic variables are obtained.

The top of the atmosphere and the Earth's surface assume rigid boundaries. Neumann boundary conditions are therefore imposed in solving the Helmholtz equation. It was further established that in the horizontal an Arakawa C-grid is used while in the vertical the Charney-Phillips grid staggering is used.

The consistency analysis of the SISL scheme showed that it is consistent with the initial partial differential equation. On the other hand, the stability analysis of the SISL scheme revealed that the linear part of the scheme is stable on condition that $\alpha_3 \geq \alpha_1 \geq \frac{1}{2}$, $\alpha_4 \geq \frac{1}{2}$, and $\alpha_2 \geq \frac{1}{2}$. By Lax Equivalence Theorem, it was established that the linear part of the scheme is convergent.

Stability analysis of the Helmholtz equation using Von Neumann method showed that stability is guaranteed when the wave number ω and the spatial step size h are chosen such that;

$$-1 < \frac{4}{4-\omega^2 h^2}\cos\frac{\pi}{m} < 1$$

Solving the Helmholtz equation iteratively, using different methods, showed that the iterations converge faster by Bi-Conjugate Gradient Stabilized Method followed by the Generalised Minimal Residual Method. The least convergent method is the Jacobi Iterative Method. While the Quasi-Minimal Residual Method converges relatively faster than the Jacobi Iterative Method, it takes longer CPU-time than all other methods. It is therefore not efficient.

The most efficient method for solving the Helmholtz equation is therefore Bi-Conjugate Gradient Stabilized Method followed by Generalised Minimal Residual Method. Finally, it was established that visualization of the Helmholtz equation can be obtained using the Graphical User Interface in MATLAB.

8.2. Recommendations

In full view of the findings in this study, we recommend that:

- The Bi-Conjugate Gradient Stabilized Method be used in the solving of the Helmholtz equation for its efficiency. This can drastically reduce the forecast time, which has been a long-standing bottleneck in Numerical Weather Prediction.
- The Generalised Minimal Residual Method be used only when, for some reasons, the Bi-Conjugate Gradient Stabilized Method cannot be adopted.
- In the choice of the spatial step for the grid, care should be taken to ensure that the condition for stability (7.1.1) is observed. Otherwise, the Helmholtz equation may be rendered unstable.

References

[1] Cullen M. J. P., Davies T., Malcolm A. J., Mawson M. H., Staniforth A., White A. A., Wood N. *A new dynamical core for the Met Office's global and regional modelling of the atmosphere; Q. J. R. Meteorological Society* (2005), **131**, pp. 1759-1782

[2] Buckeridge S., 2010, *Numerical Solution of Weather and Climate Systems,* Ph. D Thesis, University of Bath.

[3] Payne T. J., 2008, *Notes and Correspondence: A linear-stability analysis of the Semi-implicit Semi-Lagrangian Discretisation of the fully-compressible equations; Quarterly Journal of Meteorological Society,* 134:779-786 (2008), Met office, UK.

[4] Davies T. et al, 2003, *Validity of Anelastic and Other Equation sets as inferred from normal mode analysis,* Met-Office, Bracknell, UK and University of Reading, UK.

[5] Lynch P., 2007, *The Origins of Computer Weather Prediction and Climate Modelling,* Meteorology and Climate Centre, School of Mathematical Sciences, University College Dublin, Belfield, Ireland; *Journal of Computational Physics* 227(2008)3431-3444.

[6] Staniforth A., White A., Wood N., Thuburn J., Zerroukat M., Cordero E., 2004, *Joy of U.M. 6.0 – Model Formulation, Unified Model Documentation Paper No.15.* (Unpublished), Dynamics Research, Numerical Weather Prediction, Met Office, FitzRoy Road, Exeter, Devon, EX1 3PB, United Kingdom.

[7] Rizwan B., 2007, *Introduction to Numerical Analysis Using Matlab,* U.S.A.

[8] Goswami B. N., 1997, *The Challenge of Weather Prediction,* Centre for Atmospheric and Oceanic Sciences, Indian Institute of Science, Bangalore.

[9] Diamantakis M., Davies T., and Wood N., A *An Iterative time-stepping scheme for the Met Office's semi-implicit semi-Lagrangian non-hydrostatic model,* Quarterly Journal of The Royal Meteorological Society, *Q.J.Meteorol. Soc.* **133**: 997-1011(2007), Met Office, Exeter, UK.

[10] Norma A., Asnida G., Hafizah F., 2013, *Elliptic-Helmholtz Equation for early Detection of Breast Cancer Growth using MATLAB Computing,* Ibnu Sina Institute, Department of Mathematics Universiti Teknologi Malaysia.

[11] Barrett, R., *et al*, 1994, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods,* SIAM, Philadelphia PA.

[12] Golding, B., Mylne, K., Clark, P., *The History and Future of Numerical Weather Prediction in the Met Office,* 2004, Met Office, Exeter, and JCMM, Reading.

[13] Microsoft ® Encarta ® 2009. © 1993-2008 Microsoft Corporation. All rights reserved.

APPENDIX A

MATLAB CODES FOR DIFFERENT METHODS USED IN THIS PAPER

1. Jacobi Iterative Method

```
function x = JacobiM(Ab,x,acc)
%Jacobi Iterations
[n,t]=size(Ab);b=Ab(1:n,t);R=1;k=1;
d(1,1:n+1)=[0 x];while R>acc
    for i=1:n
        sum=0;
        for j=1:n; if j~=i
            sum=sum+Ab(i,j)*d(k,j+1);end;
        x(1,i)=(1/Ab(i,i))*(b(i,1)-sum);end;end
k=k+1;d(k,1:n+1)=[k-1 x];
R=max(abs((d(k,2:n+1)-d(k-1,2:n+1))));
if k>10 & R>100
    ('Jacobi Method is diverges')
     break;end;
end;x=d;
```

JacobiM(Ab,x,acc)

t = cputime; JacobiM(Ab,x,acc); e = cputime-t

2. Gauss-Seidel Iterative Method

```
function x=GaussSM(Ab,x,acc)
[n,t]=size(Ab);b=Ab(1:n,t);R=1;k=1;
d(1,1:n+1)=[0 x];k=k+1; while R>acc
    for i=1:n;sum=0; for j=1:n
        if j<=i-1; sum = sum+Ab(i,j)*d(k,j+1);
        elseif j>=i+1
            sum = sum + Ab(i,j)*d(k-1,j+1);end;end
    x(1,i)=(1/Ab(i,i))*(b(i,1)-sum);
    d(k,1)=k-1;d(k,i+1)=x(1,i);end
R=max(abs((d(k,2:n+1)-d(k-1,2:n+1))));
k=k+1; if R>100 & K>10; ('Gauss-seidel method is Diverges')
    break; end;
end;x=d;
```

GaussSM(Ab,x,acc)

t = cputime; GaussSM(Ab,x,acc); e = cputime-t

3. Successive-Over-Relaxation Method

```
function sol=SORM(Ab,w,acc)
[n,t]=size(Ab);b=Ab(1:n,t);R=1;k=1;
x=zeros(1,n);d(1,1:n+1)=[0 x];
k=k+1; while R>acc
    for i=1:n
        sum=0;
```

```
        for j=1:n
            if j<=i-1; sum=sum+Ab(i,j)*d(k,j+1);
            elseif j>=i+1; sum=sum+Ab(i,j)*d(k-1,j+1);
            end; end
        x(1,i)=(1-w)*d(k-1,i+1)+(w/Ab(i,i))*(b(i,1)-sum);
        d(k,1)=k-1;d(k,i+1)=x(1,i);end
    R=max(abs((d(k,2:n+1)-d(k-1,2:n+1))));
    if R>100 & k>10; break;end
    k=k+1;end;x=d
```

SORM(Ab,w,acc)

t = cputime; SORM(Ab,w,acc); e = cputime-t

4. Conjugate Residual Method

```
function x=CONJG(A,b,x0,acc,maxI)
x=x0;r=b+(A*x0);v=r;
alpha=r'*r; iter=0; flag=0;
normb=norm(b); if normb<eps; normb=1; end
while (norm(r)/normb>acc)
    u=A*v;t=alpha/(u'*v); x=x+t*v;
    r=r-t*u; beta=r'*r;
    v=r+beta/alpha*v; alpha=beta;
    iter=iter+1; if(iter==maxI); flag=1;
        break;end;
end;
```

CONJG(A,b,x0,acc,maxI)

t = cputime; CONJG(A,b,x0,acc,maxI); e = cputime-t

5. Bi-Conjugate Gradient Stabilized Method

```
function x1 = run_bicgstab
n = 9;
b = afun(ones(n,1));
tol = 1e-12;
maxit = 15;
x1 = bicgstab(@afun,b,tol,maxit,@mfun);

    function y = afun(x)
       y = [0; x(1:n-1)] + ...
           [((n-1)/2:-1:0)'; (1:(n-1)/2)'].*x + ...
           [x(2:n); 0];
    end

    function y = mfun(r)
        y = r ./ [((n-1)/2:-1:1)'; 1; (1:(n-1)/2)'];
    end
end
```

x = bicgstab(A,b,tol,maxit,M1)

t = cputime; x = bicgstab(A,b,tol,maxit,M1); e = cputime-t

6. Bi-Conjugate Gradient Method

```
function x1 = run_bicg
n = 9;
on = ones(n,1);
b = afun(on,'notransp');
tol = 1e-12;
maxit = 15;
M1 = spdiags([on/(-2) on],-1:0,n,n);
M2 = spdiags([4*on -on],0:1,n,n);
x1 = bicg(@afun,b,tol,maxit,M1,M2);

    function y = afun(x,transp_flag)
       if strcmp(transp_flag,'transp')      % y = A'*x
          y = 4 * x;
          y(1:n-1) = y(1:n-1) - 2 * x(2:n);
          y(2:n) = y(2:n) - x(1:n-1);
       elseif strcmp(transp_flag,'notransp') % y = A*x
          y = 4 * x;
          y(2:n) = y(2:n) - 2 * x(1:n-1);
          y(1:n-1) = y(1:n-1) - x(2:n);
       end
    end
end
```

x = bicg(A,b,tol,maxit,M1,M2)

t = cputime; x = bicg(A,b,tol,maxit,M1,M2); e = cputime-t

7. Quasi-Minimal Residual (QMR) Method

```
function x1 = run_qmr
n = 9;
on = ones(n,1);
tol = 1e-12;
maxit = 15;
M1 = spdiags([on/(-2) on],-1:0,n,n);
M2 = spdiags([4*on -on],0:1,n,n);
x1 = qmr(@afun,b,tol,maxit,M1,M2);

    function y = afun(x,transp_flag)
       if strcmp(transp_flag,'transp')      % y = A'*x
          y = 4 * x;
          y(1:n-1) = y(1:n-1) - 2 * x(2:n);
          y(2:n) = y(2:n) - x(1:n-1);
       elseif strcmp(transp_flag,'notransp') % y = A*x
          y = 4 * x;
          y(2:n) = y(2:n) - 2 * x(1:n-1);
          y(1:n-1) = y(1:n-1) - x(2:n);
```

```
        end
    end
end
```

x = qmr(A,b,tol,maxit,M1,M2)

t = cputime; x = qmr(A,b,tol,maxit,M1,M2); e = cputime-t

8. Generalised Minimal Residual (GMRES) Method

```
function x1 = run_gmres
n = 9;
b = afun(ones(n,1));
tol = 1e-12;  maxit = 15;
x1 = gmres(@afun,b,9,tol,maxit,@mfun);

    function y = afun(x)
        y = [0; x(1:n-1)] + ...
              [((n-1)/2:-1:0)'; (1:(n-1)/2)'].*x + ...
              [x(2:n); 0];
    end

    function y = mfun(r)
        y = r ./ [((n-1)/2:-1:1)'; 1; (1:(n-1)/2)'];
    end
end
```

x = gmres(A,b,9,tol,maxit,M1)

t = cputime; x = gmres(A,b,9,tol,maxit,M1); e = cputime-t