ระเบียบวิธีผลต่างอันตะอันดับสี่ชนิดกระชับ
สำหรับสมการออยเลอร์

นาย แส่ว มูนระดก

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาคณิตศาสตร์ประยุกต์
มหาวิทยาลัยเทคโนโลยีสุรนารี
ปีการศึกษา 2545
ISBN 974-533-204-6

A FOURTH-ORDER COMPACT FINITE DIFFERENCE SCHEME FOR THE EULER EQUATIONS

Mr. Seo Mounradok

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science in Applied Mathematics

Suranaree University of Technology

Academic Year 2002

ISBN 974-533-204-6

A FOURTH-ORDER COMPACT FINITE DIFFERENCE SCHEME FOR THE EULER EQUATIONS

Suranaree University of Technology has approved this thesis submitted in partial fulfillment of the requirements for a Master's Degree

แส่ว มูนระดก : ระเบียบวิธีผลต่างอันตะอันดับสี่ชนิดกระชับสำหรับสมการออยเลอร์ (A FOURTH-ORDER COMPACT FINITE DIFFERENCE SCHEME FOR THE EULER EQUATIONS) อ. ที่ปรึกษา: ASSOC. PROF. DR. NIKOLAY P. MOSHKIN, 70 หน้า. ISBN 974-533-204-6

วิทยานิพนธ์ฉบับนี้ศึกษาวิธีผลต่างอันตะสำหรับสมการออยเลอร์สองมิติที่ยุบตัวไม่ได้ในรูปแบบฟังก์ชันของการไหลชนิดที่มีความเร็วเป็นแบบหมุนวน และได้นำเสนอวิธีผลต่างอันตะอันดับสี่แบบกระชับชนิดใหม่สำหรับสมการออยเลอร์ที่เสถียร ในการสร้างผลต่างอันตะนี้ได้มีการรวมความสัมพันธ์ระหว่างสมการของฟังก์ชันของการไหลและสมการการถ่ายเทชนิดที่มีความเร็วเป็นแบบหมุนวน การทดสอบเชิงตัวเลขสำหรับสมการออยเลอร์ที่มีผลเฉลยแม่นตรงแสดงให้เห็นถึงความแม่นยำและประสิทธิภาพของวิธีผลต่างอันตะอันดับสี่ชนิดกระชับนี้

สาขาวิชาคณิตศาสตร์ ลายมือชื่อนักศึกษา [illegible]

ปีการศึกษา 2545 ลายมือชื่ออาจารย์ที่ปรึกษา [illegible]

SEO MOUNRADOK : A FOURTH-ORDER COMPACT FINITE DIFFERENCE SCHEME FOR THE EULER EQUATIONS THESIS ADVISOR: ASSOC. PROF. NIKOLAY P. MOSHKIN, Ph.D. 70 PP. ISBN 974-533-204-6

FINITE DIFFERENCE SCHEME/IDEAL-INCOMPRESSIBLE FLUID/ VORTICITY-STREAM FUNCTION/ITERATIVE METHOD

This thesis focuses on finite difference methods for the two dimensional incompressible Euler equations in the stream function-vorticity form. A new fourth-order compact scheme for the steady Euler equations is presented. The coupling relations between the stream function equation and the vorticity transport equation are exploited to construct a finite difference scheme. Numerical tests for the Euler equations with exact solutions are implemented to demonstrate the accuracy, efficiency and robustness of the fourth-order compact scheme.

School of Mathematics	Student [illegible]
Academic Year 2002	Advisor [illegible]

Acknowledgements

I would like express my sincere gratitude to my advisor Assoc. Prof. Dr. Nikolay P. Moshkin, School of Mathematics , Institute of Science, Suranaree University of Technology for his guidance and support throughout the course of this research. I have been working under his invaluable supervision for about three years and have had many unique research experiences which I will cherish throughout my life.

I am also very grateful to and would like to thank the Chairman of the School of Mathematics of Suranaree University of Technology, Assoc. Prof. Dr. Prapasri Asawakun and all the lecturers who taught and helped me during my studies at Suranaree University of Technology. They are Prof. Dr. Sergey V. Meleshko and Asst. Prof. Dr. Eckart Schulz for their help, advice and support.

I want to express my appreciation to Mr. Paichayon Siristienwatthana and all graduate students of the School of Mathematics of Suranaree University of Technology for their invaluable professional guidance and friendly encouragement.

In addition, I wish to express my special thanks to National University of Laos, the Government of the People's Republic of Laos and Asian Development Bank for offering the scholarship which has enabled me to continue my advanced studies at Suranaree University of Technology, Thailand.

Finally, I would like to thank my wife and children, Bounthane Mounradok for their love, support, encouragement, and patience throughout the sometimes tedious process of producing a dissertation. I can not thank then enough.

Seo Mounradok

Contents

Contents (Continued)

Contents (Continued)

List of Tables

List of Tables (Continued)

List of Figures

Chapter I

Introduction

1.1 Background and Objectives

Finite difference methods have long been used to approximate the solutions of ordinary or partial differential equations (PDEs). These methods involve discretizing the domain of interest into a structured mesh of grid points and approximating derivatives by difference quotients. At each grid point, the values of the desired function or functions are treated as unknowns, and the governing differential equation or equations are approximated by a sparse system of algebraic equations that can then be solved with an appropriate matrix solution algorithm. This approximation process and the resulting system properties are what distinguish one finite difference method from another.

The finite difference method is relatively straightforward process for developing numerical approximations to boundary value problems. However, it does suffer the disadvantage of requiring a structured mesh. This is in contrast to finite element methods, which are designed to accommodate highly unstructured meshes, as well as irregular problem data and solutions by virtue of finding approximations to the weak, or variational from of the problem. On the other hand, finite difference methods lend themselves quite easily to Taylor series analysis of truncation errors, and this property is exploited in the present work to develop rigorous high-order approximations.

The simplest finite difference scheme (FDS) for second order elliptic partial differential equation is the central difference scheme (CDS) on a uniform structured mesh. In this method, the first and second partial derivatives at a grid point are represented by linear combination of the three function values at, and directly adjacent to, the grid point in the corresponding coordinate directions. Higher derivatives require additional points. In general, derivatives of order p can be approximated in the CDS with $p+1$ symmetrically-located points if p is even. The coefficients for this case are also symmetric. For odd p, $p+2$ nodes are required, although the coefficients are antisymmetric, meaning the center coefficient for odd derivatives is always zero. The CDS was named to reflect this symmetry in the stencils and coefficients. Taylor series analysis of the truncation errors for central difference approximations shows that the scheme asymptotically approaches the values of the approximated derivatives at $O(h^2)$ as the mesh size h approaches zero.

Methods with accuracy better than $O(h^2)$ are called *higher-order* methods. These methods are desirable because their greater accuracy allows coarser meshes to be used, thus lowering computational costs. The standard method for achieving higher-order accuracy is to include additional grid points into the approximations representing the derivatives. In general on a structured mesh, a derivative of order p may be approximated to order m with $p+m-1$ points if p is even. If p is odd, $p+m$ points are required. A general finite difference theory was developed by J. C. Patterson (1983) for approximating first and second derivatives up to $O(h^{N-1})$ on a grid of N nodes.

High-order methods achieved in this manner always require *non-compact stencils* that utilize grid points located beyond those directly adjacent to the node about which the differences are taken. This complicates formulations near boundaries, increases matrix bandwidth, and increases communication requirements for implementation on parallel computer architectures.

In light of the problems caused by non-compact finite difference schemes, it is desirable to develop a class of schemes that are both high-order and compact. R. S. Hirsh (1975) conducted numerical experiments with a class of high-order compact (HOC) schemes in which the first and second derivatives are treated as unknowns resulting in a mixed method, unlike the schemes developed in the present work.

High-order compact (HOC) schemes of the type studied here increase the accuracy of the standard central difference approximation from $O(h^2)$ to $O(h^4)$ by including compact approximations to the leading truncation error terms. They are a spatial implementation of the temporal P. D. Lax and B. Wendroff's (1964) idea and were first proposed in the context of the present methodology by R. J. Mackinnon and G. F. Carcy (1989). R. J. Mackinnon and M. A. Langerman (1991) followed with similar research for convection diffusion problems. About the same time, S. Abarbanel and A. Kumar (1988) independently developed HOC schemes for the compressible Euler equations for outflow problem. These schemes are similar to the methods proposed by J. K. Dukowicz and J. D. Ramshaw (1979), S. C. R. Dennis and J. D. Hudson (1989) and M. M. Gupta et al. (1984), although they are derived in a different manner. They also have the additional advantage of suppressing or reducing numerical oscillations.

The early study of HOC methods was restricted to a single, steady 1D or 2D equation on uniform meshes. Analysis of these methods was also limited to their accuracy and oscillation properties. This dissertation addresses the application of HOC methods to mathematical simulation to the incompressible ideal fluid flow within a bounded domain.

The main results of the thesis are a follows:

1) A fourth-order compact finite difference scheme on the nine-point 2D stencil was derived for the stream function-vorticity formulation of the Euler equations governing the ideal incompressible fluid flow through bounded domain.

2) The new fourth-order compact schemes for the Euler equations is tested on exact solutions.

3) It is shown that the present scheme yields highly accurate numerical solutions.

1.2 Terminology

Artificial Viscosity is the adding of non-physical viscosity to a numerical approximation to help stabilize a formulation that would otherwise result in numerical oscillations.

Coarse Grids is a relative term describing computational meshes that have large grid spacing and thus a small number of grid points.

Compact Stencils are mesh stencils which only utilize grid points which lie on the mesh cells directly adjacent to the grid point at which a numerical approximation is being made. For structured grids, compact stencils are therefore restricted to 3, 3×3 and $3 \times 3 \times 3$ grid points in 1D, 2D, and 3D, respectively.

High-Order Compact (HOC) Schemes, in the context of my research, are numerical approximations which achieve high-order accuracy on a compact stencil by utilizing the governing differential equation as an auxiliary relationship to model truncation error terms.

Matrix Bandwidth refers to the size of the portion of a matrix which must be stored for certain direct solvers. Matrices whose non-zero entries all lie within a band around the matrix diagonal (which happens to be common in many approximation schemes) are called banded matrices. It is possible to store only the band and thus solve the matrix problem more efficiently. Two matrices of the same rank but different bandwidths will solve more efficiently for the matrix with the smaller bandwidth.

Non-Compact Stencils are stencils which utilize grid points outside of those which lie on a compact stencil.

Numerical Oscillations are a common problem with many numerical schemes. That is, the approximate solution will "bounce" from one cell to the next above and below the analytic solution. The causes for this are complicated, but they are related to strong convection (first derivatives having greater influence than second derivatives), coarse grids, and steep gradients in the solution (such as are found in boundary layers and shock fronts).

Sparse Matrices are matrices with a large percentage of zeros. For an iterative solver, only non-zeros need to be stored, which greatly reduces storage and time needed to solve the matrix problem.

Chapter II

General Considerations on the Euler Equations

2.1 Introduction

This chapter has an introductory nature, wherein we discuss the fundamental equation describing the motion of an incompressible nonviscous fluid and formulate some elementary properties of these equations. In our explanation we follow to the books Z. U. A. Warsi (1992), A. V. Kashikhov et al. (1990), A. J. Chorin and J. E. Marsden (1997), C. Marchioro and M. Pulvirenti (1994) where more detail available.

2.2 The Equation of Motion of an Ideal Incompressible Fluid

In this section we establish the mathematical model of an ideal incompressible fluid, deriving heuristically the equation governing its motion.

Fluid mechanics does not study the dynamics of the individual molecules constituting the fluid. We want to investigate the gross behavior of many of molecules. For this purpose, we assume the fluid as a continuum, a point of which is a very small portion of the real fluid. This small volume, a point in our mathematical description, will be called *fluid particle* or *element of fluid.*

Let D be a region in two-dimensional or three-dimensional space filled with a fluid. Let $X = (X^1, X^2, X^3)$, $X \in D$ be the coordinates of a fluid particle at time $t = 0$. Let $x = (x^1, x^2, x^3)$, $x \in D$ be the coordinates of the same fluid particle at time t. Then a fluid motion is, by definition, a function $\varphi : D \to D$

$$x = \varphi(X,t) \quad (\mathit{or} \ x^i \ = \ \varphi^i(X,t) \ ), \tag{2.1}$$

such that

a) φ is invertible,

b) φ and φ^{-1} are smooth enough so that the main operations of calculus may be performed on them,

c) $X = \varphi(X,0)$, $\varphi(X, t_1 + t_2) = \varphi(\varphi(X, t_1), t_2)$.

If X is fixed and t is changed, then equation (2.1) determines a *trajectory* of the fluid particle P initially placed at point X. On the other hand side, if t is fixed, then equation (2.1) determines the transformation of the fluid domain at time $t = 0$ to the fluid domain at $t = t_1$.

In spite of the fact that equation (2.1) determines the fluid motion, it is also important to study the time evolution at a given point $x \in D$ of the *density field* $\rho = \rho(x,t)$, *velocity field* $\vec{u} = \vec{u}(x,t)$ and so on.

The derivation of the fluid motion equation is based on three conservation principles

I) mass is neither created nor destroyed;

II) the rate of change of momentum of a portion of the fluid equals the force applied to it *(Newton's second law);*

III) energy is neither created nor destroyed.

A very useful concept in fluid dynamics is that of a "material volume".

Definition (see for example Z. U. A. Warsi (1992))

A *material volume* is an arbitrary collection of fluid of fixed identity and enclosed by a surface also formed a fluid particles.

All points of material volume, including the points of its boundary, move with the local continuum velocity. A material volume moves with the flow and deforms in shape as the flow progresses, with the stipulation that no mass ever fluxes in or out of the volume, viz., both the volume and its boundary are always composed of the same fluid particles. We shall denote a material volume by $V(t)$ and its surface by $S(t)$. Note that the use of material volume in fluid dynamics is in the form of a thought experiment in which one isolates a parcel of fluid out of the flow field and gives it a hypothetical surface. This helps formulate the conservation laws for fluid dynamics in a straightforward manner.

Since there can be no mass transfer to or from a material volume, the principle of mass conservation implies that

$$\frac{d}{dt}\int_{V(t)} \rho(r,t)dV = 0, \tag{2.2}$$

where $\frac{d}{dt}$ is the total time derivative , $V(t)$ is a material volume.

To transform equation (2.2), we use

$$\frac{d}{dt}\int_{V(t)} F(r,t)dV = \int_{V(t)} (\frac{DF}{Dt} + F div\,\vec{u})dV,$$

and Euler's formula

$$\frac{dJ}{dt} = J\,div\,\vec{u},$$

where J is the Jacobian of (2.1) and $J = det(\frac{\partial X_i}{\partial x_i}) \neq 0$.

With $F = \rho$, so that

$$\int_{V(t)} (\frac{D\rho}{Dt} + \rho\,div\,\vec{u})dV = 0. \tag{2.3}$$

The result in equation (2.3) is valid for any choice of the volume $V(t)$, therefore

$$\frac{D\rho}{Dt} + \rho\,div\,\vec{u} = 0, \tag{2.4}$$

which is called the differential equation for the conservation of mass, or simply the *equation of continuity*. Using definition of $\frac{D}{Dt} = \frac{\partial}{\partial t} + \vec{u} \cdot grad$, we can also write equation (2.4) as

$$\frac{\partial \rho}{\partial t} + div(\rho \vec{u}) = 0. \tag{2.5}$$

In the case of steady flow, the equation of continuity (2.5) takes the form

$$div(\rho \vec{u}) = 0. \tag{2.6}$$

Fluid flows for which the density remains constant are termed *incompressible* flows. The differential form of the continuity equation then is simply

$$div \vec{u} = 0. \tag{2.7}$$

Note that equation (2.7) is applicable to both the steady and nonsteady incompressible flows.

Let us define an *ideal fluid* as one with the following property: for any motion of the fluid, there is a function $P(x,t)$ called the *pressure* such that if S is a surface in the fluid with a chosen unit normal $\vec{n}$, the force of stress exerted across the surface S per unit area at $x \in S$ at time t is $P(x,t) \cdot \vec{n}$. By Newton's second law (force=mass $\otimes$ acceleration), we obtain the differential equation of the law of balance of momentum

$$\rho(\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u}) = - \nabla P + \rho f, \tag{2.8}$$

where $f = f(x,t)$ is external force per unity volume. Equation (2.8), together with the equation (2.6) form the *Euler equations* for an ideal incompressible fluid.

Remarks:

1. In the present thesis, we will assume the density to be always constant (for simplicity $\rho \equiv 1$),

2. When f is a potential force, ($f = -\bigtriangledown U$ for the some scalar field U), we can modify equation (2.8) to

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \bigtriangledown)\vec{u} = -\bigtriangledown H, \tag{2.9}$$

with H replaced by $P + U$.

3. Later on, we will assume the absence of external forces.

2.3 Vorticity and Stream Function

A fundamental concept of fluid motion analysis is the concept of the *vorticity field* $\vec{\omega}(x)$. By definition (see for example Z. U. A. Warsi (1992))

$$\vec{\omega} \equiv curl\,\vec{u} = \bigtriangledown \times \vec{u}. \tag{2.10}$$

The vorticity field $\vec{\omega}(x)$ gives a measure of how the fluid is rotating. The vorticity field is an important tool in studying the behavior of fluids. The Euler equations can be expressed in terms of vorticity. Using the following vector identity

$$\frac{1}{2} \bigtriangledown |\vec{u}|^2 = \vec{u} \times curl\,\vec{u} + (\vec{u} \cdot \bigtriangledown)\vec{u}, \tag{2.11}$$

the Euler equations can be written as

$$\frac{\partial \vec{u}}{\partial t} + \frac{1}{2} \bigtriangledown |\vec{u}|^2 - \vec{u} \times curl\,\vec{u} = -\bigtriangledown P.$$

Taking the curl of both sides

$$\frac{\partial \vec{\omega}}{\partial t} - curl\,(\vec{u} \times \vec{\omega}) = 0.$$

Since

$$curl\,(\vec{u} \times \vec{\omega}) = (\vec{\omega} \cdot \bigtriangledown)\vec{u} - \vec{\omega}(\bigtriangledown \cdot \vec{u}) - (\vec{u} \cdot \bigtriangledown)\vec{\omega} + \vec{u}(\bigtriangledown \cdot \vec{\omega}).$$

We finally obtain

$$\frac{\partial \vec{\omega}}{\partial t} + (\vec{u} \cdot \nabla)\vec{\omega} = (\vec{\omega} \cdot \nabla)\vec{u}. \tag{2.12}$$

However, to study the Euler equations in (2.12), it is necessary to reconstruct the velocity field $\vec{u}$ from the vorticity. In other words, we have to solve the following equation in the unknown quantity $\vec{u}$

$$curl\,\vec{u} = \vec{\omega},$$

$$\nabla \cdot \vec{u} = 0.$$

In two dimensions (2.12) becomes much simpler. Namely, in the presence of a planar symmetry

$$\vec{u} = (u_1, u_2, 0) = (u, v, 0), \quad u_i = u_i(x_1, x_2),$$

$$\vec{\omega} = (\omega_1, \omega_2, \omega_3) = (0, 0, \omega),$$

only the third component of the vorticity $\vec{\omega}$ is present $\vec{\omega} = (0, 0, \omega)$ and the right-hand side of (2.12) vanishes. Therefore, the Euler equation for the vorticity in two-dimensions becomes

$$\frac{\partial \omega}{\partial t} + (\vec{u} \cdot \nabla)\omega = 0. \tag{2.13}$$

Notice that (2.13) implies the conservation of the vorticity along the trajectories.

For two-dimensional steady or nonsteady incompressible flow, the equation of continuity is

$$div\,\vec{u} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0. \tag{2.14}$$

Then it is obvious that if we take an arbitrary function $\psi(x, y)$ and derive u, v according to the rules

$$u = \frac{\partial \psi}{\partial y}, \qquad v = -\frac{\partial \psi}{\partial x}, \tag{2.15}$$

equation (2.14) will be satisfied identically. Such a function ψ is call a *stream function.*

2.4 Initial Boundary Value Problem for the Euler Equation

The Euler equations can be interpreted as the limit of the Navier-Stokes equations of vanishing viscosity, and their character is hyperbolic in the unsteady cases. The Euler equations governing the flow of an incompressible fluid of null viscosity, called also ideal fluid, are

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u} = -\nabla p, \tag{2.16}$$

$$\text{div } \vec{u} = 0, \tag{2.17}$$

where $\vec{u} = \vec{u}(\vec{x}, t)$ is the velocity vector, $\vec{x} = x(x_1, x_2, x_3)$ are the Cartesian coordinates of a point, $p(\vec{x}, t)$ is the pressure divided by the constant density of the fluid. If $\vec{u} = (u_1, u_2, u_3)$

$$\begin{aligned} \nabla &\equiv (\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \frac{\partial}{\partial x_3}) \quad \textit{is the gradient}, \\ div(\vec{u}) &= \frac{\partial u_1}{\partial x_1} + \frac{\partial u_2}{\partial x_2} + \frac{\partial u_3}{\partial x_3} \quad \textit{is the divergence.} \end{aligned}$$

The statement of the problem is made complete by the specification of suitable boundary and initial conditions. A typical boundary condition consists in prescribing the value of the *normal* component of velocity b_n on the boundary

$$\vec{n} \cdot \vec{u} \mid_S = b_n(x_S, t), \tag{2.18}$$

where $\vec{n}$ denotes the outward unit normal to the boundary S of domain V and $x_S \in S$. The tangential components of velocity on the boundary need not be specified in the nonviscous problem.

The initial condition consists in the specification of the velocity field $\vec{u}_0$ at the initial time, $t = 0$, namely,

$$\vec{u}\mid_{t=0} = \vec{u}_0(x). \tag{2.19}$$

The boundary value of the normal component of velocity b_n must satisfy, for all $t \geq 0$, the global condition

$$\oint_S b_n dS = 0, \tag{2.20}$$

which follows from integrating the continuity equation over V and using the divergence theorem. Moreover, the initial velocity field $\vec{u}_0$ is assumed to be solenoidal, *i.e.*,

$$\nabla \cdot \vec{u}_0 = 0. \tag{2.21}$$

Finally, the boundary and initial data b_n and $\vec{u}_0$ are assumed to satisfy the following compatibility condition

$$b_n \mid_{t=0} = \vec{n} \cdot \vec{u}_0|_S. \tag{2.22}$$

This compatibility condition between the boundary and initial values is necessary to prove the existence and uniqueness of classical solutions of the problem in two dimensions.

The boundary condition (2.18) is associated with the "elliptic character" of the incompressibility. If $b_n = 0$, then no other boundary condition is requested. On the contrary, if $b_n \neq 0$ on some part of S, then the fluid enters the volume V or is flowing out of it and some other boundary condition must be imposed to comply with the "hyperbolic" character of the convective part of the problem.

2.5 Euler Equations in the Cartesian Coordinate System

Let $u_i = u_i(x, y, z, t),\ i = 1, 2, 3,\ p = p(x, y, z, t)$ and x, y, z be Cartesian coordinates. Equations (2.16) and (2.17) can be written in componentwise form

$$\frac{\partial u_1}{\partial t} + u_1\frac{\partial u_1}{\partial x} + u_2\frac{\partial u_1}{\partial y} + u_3\frac{\partial u_1}{\partial z} = -\frac{\partial p}{\partial x}, \tag{2.23}$$

$$\frac{\partial u_2}{\partial t} + u_1\frac{\partial u_2}{\partial x} + u_2\frac{\partial u_2}{\partial y} + u_3\frac{\partial u_2}{\partial z} = -\frac{\partial p}{\partial y}, \tag{2.24}$$

$$\frac{\partial u_3}{\partial t} + u_1\frac{\partial u_3}{\partial x} + u_2\frac{\partial u_3}{\partial y} + u_3\frac{\partial u_3}{\partial z} = -\frac{\partial p}{\partial z}, \tag{2.25}$$

$$\frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial y} + \frac{\partial u_3}{\partial z} = 0. \tag{2.26}$$

Let $u_i = u_i(x, y, t),\ i = 1, 2,\ u_3 = 0,\ p = p(x, y, t)$ and x, y be Cartesian coordinates. Equations (2.16) and (2.17) can be written in componentwise form

$$\frac{\partial u_1}{\partial t} + u_1\frac{\partial u_1}{\partial x} + u_2\frac{\partial u_1}{\partial y} = -\frac{\partial p}{\partial x}, \tag{2.27}$$

$$\frac{\partial u_2}{\partial t} + u_1\frac{\partial u_2}{\partial x} + u_2\frac{\partial u_2}{\partial y} = -\frac{\partial p}{\partial y}, \tag{2.28}$$

$$\frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial y} = 0. \tag{2.29}$$

For a velocity field $\vec{u} = u_1\vec{i} + u_2\vec{j}$, where $\vec{i}$ and $\vec{j}$ are unit vectors in the x and y directions, respectively, the stream function ψ may be defined to within an arbitrary constant by

$$u_1 = \frac{\partial \psi}{\partial y}, \tag{2.30}$$

$$u_2 = -\frac{\partial \psi}{\partial x}. \tag{2.31}$$

The definition of ψ implies that the incompressible continuity condition is satisfied. The 2D scalar vorticity is defined as the signed magnitude of the curl of the velocity, $\omega = \frac{\partial u_2}{\partial x} - \frac{\partial u_1}{\partial y}$, and (2.30) and (2.31) imply that

$$\nabla^2 u = \psi_{xx} + \psi_{yy} = -\omega. \tag{2.32}$$

Differentiate equation (2.27) with respect to y and differentiate equation (2.28) with respect to x, subtract first from the second, we obtain

$$\psi_y \omega_x - \psi_x \omega_y = 0. \tag{2.33}$$

The Euler equations representing the steady-state two-dimensional incompressible fluid flow given in stream function-vorticity form are equations (2.32) and (2.33) where ψ and ω represent the stream function and vorticity vector, respectively.

Chapter III

High-Order Compact Methodology

3.1 Introduction

This chapter provides an overview of high-order compact theory as applied to the steady convection diffusion equation. It covers 1D two-point boundary value problems and 2D boundary value problems for the Poisson equation solved on uniform grids.

3.2 High-Order Compact Methodology for One-Dimensional Problem

To introduce the basic ideas, let us consider a one-dimensional two-point boundary-value problem with governing equation

$$-au'' + bu' + cu = f, \quad on\, \Omega \;=\; \big(0, 1\big), \tag{3.1}$$

where f is a given function, prime ($'$) denotes differentiation with respect to x, the coefficients a, b and c are constant, and standard Dirichlet boundary conditions hold at $x = 0$ and $x = 1$. We may introduce the flux $\sigma = -au'$ to rewrite (3.1) as the first order system

$$\sigma' - \frac{b}{a}\sigma + cu = f, \tag{3.2}$$

$$\sigma + au' = 0. \tag{3.3}$$

Central differencing this pair of equations (3.2) and (3.3) on a uniform grid with mesh size h yields the discrete relations for the exact nodal values u_i and σ_i at $x = x_i$

$$\delta_x \sigma_i - \frac{b}{a}\sigma_i + cu_i = f_i + \frac{h^2}{6}\sigma_i''' + O(h^4), \tag{3.4}$$

$$\sigma_i + a\delta_x u_i = a\frac{h^2}{6}u_i''' + O(h^4), \tag{3.5}$$

where δ_x denotes the standard central difference approximation to $\frac{d}{dx}$ and $u_i''' = \frac{d^3u}{dx^3}$ at $x = x_i$

$$\delta_x u_i = \frac{u_{i+1} - u_i}{2h} = u_i' + \frac{h^2}{6}u_i''' + O(h^4).$$

Assuming f is sufficiently regular we can differentiate equation (3.2) with respect to x to obtain the auxiliary relation

$$\sigma'' = f' + \frac{b}{a}\sigma' - cu', \tag{3.6}$$

and again differentiate to get

$$\sigma''' = f'' + \frac{b}{a}\sigma'' - cu'',$$

Then

$$\sigma_i''' = f_i'' + \frac{b}{a}\sigma_i'' - cu_i'',$$

which can be approximated to $O(h^2)$ as

$$\sigma_i''' = f_i'' + \frac{b}{a}\delta_x^2\sigma_i - c\delta_x^2 u_i + O(h^2), \tag{3.7}$$

where δ_x^2 denotes the standard central difference operator for $\frac{d^2}{dx^2}$

$$\delta_x^2 u_i = \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + O(h^2).$$

Since σ''' in equation (3.4) is scaled by h^2, the resulting approximation in equation (3.4) remains $O(h^4)$ upon substitution of equation (3.7). Note also that an alternative exact representation can be derived as follows:

First use equations (3.2) and (3.3) in equation (3.6) to obtain

$$\sigma'' = f' + \frac{b}{a}(f + \frac{b}{a}\sigma - cu) + \frac{c}{a}\sigma.$$

Differentiating once more,

$$\sigma''' = f'' + \frac{b}{a}(f' + \frac{b}{a}\sigma' - cu') + \frac{c}{a}\sigma'.$$

Applying equations (3.2) and (3.3) at x_i again,

$$\sigma_i''' = f_i'' + \frac{b}{a}(f_i' + \frac{b}{a}\chi_i + \frac{c}{a}\sigma_i) + \frac{c}{a}\chi_i, \tag{3.8}$$

where $\chi_i = f_i + \frac{b}{a}\sigma_i - cu_i$. Hence we can use either equation (3.7) or equation (3.8) for σ_i''' in equation (3.4) to obtain an accurate compact scheme.

A similar strategy can be developed for u_i''' in equation (3.5). More specifically, from equation (3.3), $u' = -\frac{\sigma}{a}$ so $u''' = -\frac{\sigma''}{a}$ implies

$$u_i''' = -\frac{1}{a}\delta_x^2\sigma_i + O(h^2), \tag{3.9}$$

or

$$u'' = -\frac{1}{a}\sigma' = -\frac{1}{a}(f + \frac{b}{a}\sigma - cu),$$

and hence the exact result is

$$u_i''' = -\frac{1}{a}(f_i' + \frac{b}{a}\chi_i + \frac{c}{a}\sigma_i), \tag{3.10}$$

where $\chi_i = f_i + \frac{b}{a}\sigma_i - cu_i$. Using the relations (3.9) or (3.10) for u''' in equation (3.5) yields a higher-order compact mixed approximation that is $O(h^4)$ accurate at the grid points. For example, the resulting discrete $O(h^4)$ approximation at an interior point i using the relations (3.7) and (3.9) in equations (3.4) and (3.5) becomes

$$\delta_x\sigma_i - \frac{b}{a}\sigma_i + cu_i - \frac{h^2}{6}(f_i'' + \frac{b}{a}\delta_x^2\sigma_i - c\delta_x^2u_i) = f_i\,,$$

$$\sigma_i + a\delta_x u_i + \frac{h^2}{6}\delta_x^2\sigma_i = 0\;.$$

Regrouping terms,

$$\left(\delta_x\sigma_i - \frac{b}{a}\sigma_i - \frac{bh^2}{6a}\delta_x^2\sigma_i\right) + \left(cu_i + \frac{ch^2}{6}\delta_x^2u_i\right) = f_i + \frac{h^2}{6}f_i'', \tag{3.11}$$

$$\left(\sigma_i + \frac{h^2}{6}\delta_x^2\sigma_i\right) + a\delta_x u_i = 0, \tag{3.12}$$

where δ_x, δ_x^2 denote the respective first and second central differences. If f'' is known analytically, it can be used directly in equation (3.11). If not, the approximation $\delta_x^2 f_i$ is acceptable, because f_i'' is scaled by h^2, thus the overall truncation error remains $O(h^4)$.

We remark that all three types of boundary conditions are easily accommodated exactly since the values of both u and σ enter explicitly at the end nodes. High-order compact expressions for non-Dirichlet boundary conditions are not trivial for direct approximation of equation (3.1).

If the exact representations in equations (3.8) and (3.10) are used, we again have an $O(h^4)$ scheme here. However, it is clear that this approach can now be continued to obtain similar representations for the $O(h^4)$ terms and thereby generate a higher-order compact scheme that is $O(h^6)$ accurate.

Example (HOC that is $O(h^6)$)

Central differencing of equations (3.2) and (3.3) on a uniform grid with mesh size h yields the discrete relations for the exact nodal values u_i and σ_i at $x = x_i$

$$\delta_x \sigma_i - \frac{b}{a}\sigma_i + cu_i = f_i + \frac{h^2}{6}\sigma_i''' + \frac{h^4}{120}\sigma_i^V + O(h^6), \tag{3.13}$$

$$\sigma_i + a\delta_x u_i = \frac{ah^2}{6}u_i''' + \frac{ah^4}{120}u_i^V + O(h^6), \tag{3.14}$$

where δ_x denotes the standard central difference approximation to $\frac{d}{dx}$, $u_i''' = \frac{d^3u}{dx^3}$ and $u_i^V = \frac{d^5u}{dx^5}$ at $x = x_i$

$$\delta_x u_i = \frac{u_{i+1} - u_i}{2h} = u_i' + \frac{h^2}{6}u_i''' + \frac{h^4}{120}u_i^V + O(h^6).$$

Differentiating equation (3.2) four-times with respect to x and applying equations (3.2) and (3.3) at x_i again,

$$\begin{aligned} \sigma_i^V = f_i^{IV} &+ \frac{b}{a}\left(f_i''' + \frac{b}{a}(f_i'' + \frac{b}{a}\Phi_i + \frac{c}{a}\chi_i) + \frac{c}{a}\Phi_i\right) \\ &+ \frac{c}{a}(f_i'' + \frac{b}{a}\Phi_i + \frac{c}{a}\chi_i), \end{aligned} \tag{3.15}$$

where $\Phi_i = f_i' + \frac{b}{a}\chi_i + \frac{c}{a}\sigma_i$ and $\chi_i = f_i + \frac{b}{a}\sigma_i - cu_i$. Substituting equations (3.8) and (3.15) into equation (3.13), we obtain

$$\begin{aligned} \delta_x \sigma_i - \frac{b}{a}\sigma_i + cu_i = f_i &+ \frac{h^2}{6}\Psi_i \\ &+ \frac{h^4}{120}\left(f_i^{IV} + \frac{b}{a}(f_i''' + \frac{b}{a}\Psi_i + \frac{c}{a}\Phi_i) + \frac{c}{a}\Psi_i\right) \\ &+ O(h^6), \end{aligned} \tag{3.16}$$

where $\Psi_i = f_i'' + \frac{b}{a}\Phi_i + \frac{c}{a}\chi_i$, $\Phi_i = f_i' + \frac{b}{a}\chi_i + \frac{c}{a}\sigma_i$ and $\chi_i = f_i + \frac{b}{a}\sigma_i - cu_i$. Differentiating equation (3.3) four-times with respect to x and applying equations (3.2) and (3.3) at x_i again,

$$u_i^V = -\frac{1}{a}(f_i''' + \frac{b}{a}\Psi_i + \frac{c}{a}\Phi_i), \tag{3.17}$$

where $\Psi_i = f_i'' + \frac{b}{a}\Phi_i + \frac{c}{a}\chi_i$, $\Phi_i = f_i' + \frac{b}{a}\chi_i + \frac{c}{a}\sigma_i$ and $\chi_i = f_i + \frac{b}{a}\sigma_i - cu_i$. Substituting equations (3.10) and (3.17) into equation (3.14), we obtain

$$\sigma_i + a\delta_x u_i = -\frac{h^2}{6}\Phi_i - \frac{h^4}{120}(f_i''' + \frac{b}{a}\Psi_i + \frac{c}{a}\Phi_i) + O(h^6), \tag{3.18}$$

where $\Psi_i = f_i'' + \frac{b}{a}\Phi_i + \frac{c}{a}\chi_i$, $\Phi_i = f_i' + \frac{b}{a}\chi_i + \frac{c}{a}\sigma_i$ and $\chi_i = f_i + \frac{b}{a}\sigma_i - cu_i$. In fact, this approach can be generalized to any order. Moreover, it is possible to develop the scheme for more general situations such as non-constant coefficients.

3.3 High-Order Compact Finite Difference Schemes for Two-Dimensional Equation.

We consider the more interesting and practical extension to the model elliptical partial differential equation in two dimensions.

3.3.1 Statement of the Problem.

We will be primarily concerned with second-order elliptic equations in two-dimensional domain. The commonly occurring elliptic equation is the **Poisson equation** given by

$$u_{xx} + u_{yy} = f(x,y),\ (x,y) \in \Omega,\ \Omega \subset \mathbb{R}^2, \tag{3.19}$$

where $u_{xx} = \frac{\partial^2 u}{\partial x^2}$ and $u_{yy} = \frac{\partial^2 u}{\partial y^2}$.

Using the operator $\triangle = \nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$, the Poisson equation is written as $\triangle u = f$. The operator $\triangle$ is called the **Laplacian** operator. The homogeneous equation corresponding to equation (3.19), called the **Laplace equation**, is given by

$$\triangle u = 0 \ \text{ in } \ \Omega.$$

In order to completely determine the solution of equation (3.19) it is necessary to specify a boundary condition on the solution $u(x,y)$. The boundary condition is

$$u(x,y) = b_1(x,y),\ (x,y)\ \in\ \partial\Omega, \tag{3.20}$$

where $b_1(x, y)$ is a given function. Boundary condition (3.20) is called **Dirichlet boundary condition**.

Let us consider the Dirichlet problem for the Poisson equation (3.19) in the unit square in the (x, y) plane. To be specific, we wish to solve the Poisson equation

$$u_{xx} + u_{yy} = f(x, y) \text{ in } \Omega = \{0 < x, y < 1\}, \tag{3.21}$$

with the Dirichlet boundary condition

$$u(x, y) = g(x, y) \text{ on } \partial\Omega. \tag{3.22}$$

In other words, we wish to obtain $u(x, y)$ for the point (x, y) within Ω, given $f(x, y)$ and $g(x, y)$. In order to solve equations (3.21) and (3.22) numerically, we first impose a grid structure on Ω. This is accomplished as follows. For an integer $N > 1$, define

$$\Omega_h = \{(x_i, y_j) = (ih, jh) \text{ for } i, j = 0, 1, ..., N; h = \frac{1}{N}\}. \tag{3.23}$$

Figure 3.1 shows the grid corresponding to $N = 5$.

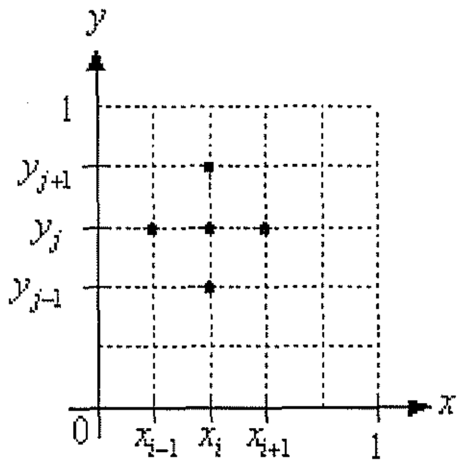

Figure 3.1: Grid structure for 2D Poisson equation

3.3.2 Second-Order Compact Finite Difference Scheme

By the standard central difference approximation to $\frac{\partial^2 u}{\partial x^2}$ and $\frac{\partial^2 u}{\partial y^2}$ at $x = x_i$, and $y = y_i$ we obtain

$$u_{xx}(x_i, y_j) = \frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j)}{h^2} + O(h^2),$$

$$u_{yy}(x_i, y_j) = \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1})}{h^2} + O(h^2).$$

Dropping the truncation error terms in each of the equations above, and letting $v_{i,j}$ denote the approximation to $u(x_i, y_j)$, we obtain a difference scheme for the partial differential equation (3.21). For $1 \leq i, j \leq N-1$, we have

$$\frac{v_{i+1,j} - 2v_{i,j} + v_{i-1,j}}{h^2} + \frac{v_{i,j+1} - 2v_{i,j} + v_{i,j-1}}{h^2} = f_{i,j}, \tag{3.24}$$

where $f_{i,j} = f(x_i, y_j)$. At each interior grid point (x_i, y_j), the PDE is discretized by using the five grid points

$$(x_{i-1}, y_j), (x_i, y_{j-1}), (x_i, y_j), (x_i, y_{j+1}), (x_{i+1}, y_j).$$

These grid points are highlighted by dots in Figure 3.1. For this reason, the difference equation (3.24) is frequently referred to as five-point Laplacian discretization. The boundary condition (3.22) allows us to specify

$$u(x_0, y_j) \text{ and } u(x_N, y_j) \text{ for } j = 0, 1, ..., N,$$

and

$$u(x_i, y_0) \text{ and } u(x_i, y_N) \text{ for } i = 0, 1, ..., N-1.$$

Multiplying throughout by h^2 in equation (3.24) and simplifying results in

$$4v_{i,j} - v_{i+1,j} - v_{i-1,j} - v_{i,j+1} - v_{i,j-1} = -h^2 f_{i,j}, \tag{3.25}$$

for $i, j = 1, 2, ..., N-1$. Note that equation (3.25) is a linear system in $(N-1)^2$ unknowns. Thus the coefficient matrix for this system is of order $(N-1)^2 \times (N-1)^2$. Figure 3.2 shows the labelling of the grid points corresponding to the equations. At most five unknowns appear in each equation. This results in a

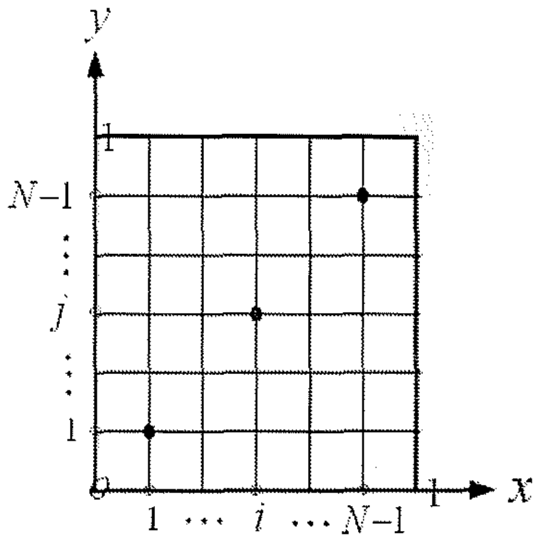

Figure 3.2: Labelling of grid points.

very sparse linear system. In particular, with the labelling scheme we are using, the linear system may be described by $Au = f$ and the coefficient matrix A has a symmetric block tridiagonal form. Example: For $n = 5$ the matrix A has the form

$$\mathbf{A} = \begin{pmatrix} B & -I & 0 & 0 \\ -I & B & -I & 0 \\ 0 & -I & B & -I \\ 0 & 0 & -I & B \end{pmatrix},$$

in which each I is a 4×4 identity matrix and each B is a 4×4 tridiagonal matrix of the form

$$\mathbf{B} = \begin{pmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{pmatrix}$$

where

$$u = \begin{pmatrix} u_{11} \\ u_{12} \\ \vdots \\ u_{44} \end{pmatrix} \qquad \textit{and} \qquad f = -h^2 \begin{pmatrix} f_{11} \\ f_{12} \\ \vdots \\ f_{44} \end{pmatrix}$$

There are two approaches to the solution to $Ax = b$. One is to pick an appropriate direct method and adapt it to the linear system. In contrast to the direct methods are the iterative methods. These methods generate a sequence of approximate solution $\{x^{(k)}\}_{k=1}^{\infty}$ and essentially involve the matrix A only in the context of matrix-vector multiplication.(see more details in Appendix B)

For the purpose of describing the iterative methods, we rewrite equation (3.25) as

$$v_{i,j} = \frac{1}{4}\big(v_{i-1,j} + v_{i+1,j} + v_{i,j-1} + v_{i,j+1} - h^2 f_{i,j}\big), \tag{3.26}$$

where we use notations $f_{i,j} = f(x_i, y_j)$ and $v_{i,j} = u(x_i, y_j)$. Then the main iterative methods are

$$v_{i,j}^{(k+1)} = \frac{1}{4}\big(v_{i-1,j}^{(k)} + v_{i+1,j}^{(k)} + v_{i,j-1}^{(k)} + v_{i,j+1}^{(k)} - h^2 f_{i,j}\big), \tag{3.27}$$

$$v_{i,j}^{(k+1)} = \frac{1}{4}\big(v_{i-1,j}^{(k+1)} + v_{i+1,j}^{(k)} + v_{i,j-1}^{(k+1)} + v_{i,j+1}^{(k)} - h^2 f_{i,j}\big), \tag{3.28}$$

$$v_{i,j}^{(k+1)} = \frac{\omega}{4}\big(v_{i-1,j}^{(k+1)} + v_{i+1,j}^{(k)} + v_{i,j-1}^{(k+1)} + v_{i,j+1}^{(k)} - h^2 f_{i,j}\big) + (1-\omega) v_{i,j}^{(k)}. \tag{3.29}$$

Note that equation (3.27) is the Jacobi iterative method, equation (3.28) is the Gauss-Seidel iterative method, and equation (3.29) is the SOR iterative method. Recall that each of these methods requires an initial guess $v_{i,j}^{(0)}$ for the unknowns.

3.3.3 Fourth-Order Compact Finite Difference Scheme

In this section we formulate a compact fourth-order finite difference method that can solve the boundary value problem (3.21) and (3.22) with the novelty of "genuine compactness", i.e. the compact scheme is strictly within the nine-point stencil.

To set up a compact finite difference scheme for equation (3.21), we use Figure 3.3 which shows the placement of the nine-points. Assuming a uniform grid in both x and y-directions, we number the mesh points (x, y), $(x+h, y)$, $(x, y+h)$, $(x-h, y)$, $(x, y-h)$, $(x+h, y+h)$, $(x-h, y+h)$, $(x-h, y-h)$ and $(x+h, y-h)$ as $0, 1, 2, 3, 4, 5, 6, 7$ and 8, respectively, where h is the grid size. In writing the FD approximations, a single subscript "j" denotes the corresponding function value at the mesh point numbered "j".

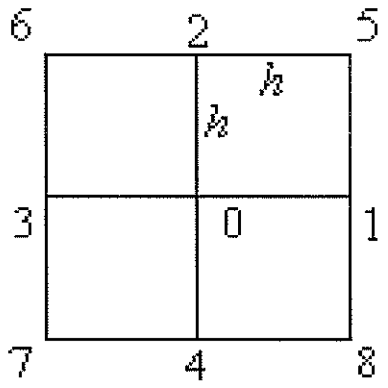

Figure 3.3: Computational stencil

Following (R. S. Hirsh (1975)), the second derivatives at the point (x, y)

are approximated by

$$\delta_x^2 u_0 = \frac{u_3 - 2u_0 + u_1}{h^2} = (u_{xx})_0 + \frac{h^2}{12}(u_{xxxx})_0 + O(h^4), \tag{3.30}$$

and

$$(u_{xxxx})_0 = \frac{(u_{xx})_3 - 2(u_{xx})_0 + (u_{xx})_1}{h^2} + O(h^2) = \delta_x^2(u_{xx})_0 + O(h^2). \tag{3.31}$$

Substituting equation (3.31) into equation (3.30), we obtain

$$\delta_x^2 u_0 = (u_{xx})_0 + \frac{h^2}{12}\delta_x^2(u_{xx})_0 + O(h^4),$$

or

$$\delta_x^2 u_0 = \big(1 + \frac{h^2}{12}\delta_x^2\big)(u_{xx})_0 + O(h^4),$$

or

$$(u_{xx})_0 = \big((1 + \frac{h^2}{12}\delta_x^2)^{-1}\delta_x^2\big)u_0 + O(h^4). \tag{3.32}$$

A similarly for $(u_{yy})_0$, we obtain

$$(u_{yy})_0 = \big((1 + \frac{h^2}{12}\delta_y^2)^{-1}\delta_y^2\big)u_0 + O(h^4), \tag{3.33}$$

where the operators δ_x^2 and δ_y^2 are defined as

$$\delta_x^2 u_0 = \frac{u_1 - 2u_0 + u_3}{h^2}, \quad \delta_y^2 u_0 = \frac{u_2 - 2u_0 + u_4}{h^2}. \tag{3.34}$$

Substituting equations (3.32) and (3.33) into the left-hand side of equation (3.19) yields

$$\left((1 + \frac{h^2}{12}\delta_x^2)^{-1}\delta_x^2\right)u_0 + \left((1 + \frac{h^2}{12}\delta_y^2)^{-1}\delta_y^2\right)u_0 = f_0 + O(h^4).$$

Applying the operator $\big(1 + \frac{h^2}{12}\delta_x^2\big)\big(1 + \frac{h^2}{12}\delta_y^2\big) = \big(1 + \frac{h^2}{12}\delta_y^2\big)\big(1 + \frac{h^2}{12}\delta_x^2\big)$, we get

$$\left((1 + \frac{h^2}{12}\delta_y^2)\delta_x^2\right)u_0 + \left((1 + \frac{h^2}{12}\delta_x^2)\delta_y^2\right)u_0 = f_0\big(1 + \frac{h^2}{12}\delta_y^2\big)\big(1 + \frac{h^2}{12}\delta_x^2\big) + O(h^4),$$

or

$$\Big((1+\frac{h^2}{12}\delta_y^2)\delta_x^2\Big)u_0+\Big((1+\frac{h^2}{12}\delta_x^2)\delta_y^2\Big)u_0=(1+\frac{h^2}{12}\delta_x^2+\frac{h^2}{12}\delta_y^2)f_0+O(h^4). \tag{3.35}$$

Using equation (3.34) in both sides of equation (3.35), we get a fourth-order compact finite difference scheme for the Poisson-type equation given by equation (3.21)

$$\delta_x^2u_0+\frac{h^2}{12}\delta_y^2(\frac{u_3-2u_0+u_1}{h^2})+\delta_y^2u_0+\frac{h^2}{12}\delta_x^2(\frac{u_2-2u_0+u_4}{h^2})=$$
$$(f_0+\frac{h^2}{12}\delta_x^2f_0+\frac{h^2}{12}\delta_y^2f_0)+O(h^4),$$

or

$$\delta_x^2u_0+\frac{1}{12}(\delta_y^2u_3-2\delta_y^2u_0+\delta_y^2u_1)+\delta_y^2u_0+\frac{1}{12}(\delta_x^2u_2-2\delta_x^2u_0+\delta_x^2u_4)=$$
$$(f_0+\frac{h^2}{12}\delta_x^2f_0+\frac{h^2}{12}\delta_y^2f_0)+O(h^4),$$

or

$$\frac{u_3-2u_0+u_1}{h^2}+\frac{1}{12}(\frac{u_6-2u_3+u_7}{h^2}-2\frac{u_2-2u_0+u_4}{h^2}+\frac{u_5-2u_1+u_8}{h^2})$$
$$+\frac{u_2-2u_0+u_4}{h^2}+\frac{1}{12}(\frac{u_6-2u_2+u_5}{h^2}-2\frac{u_3-2u_0+u_1}{h^2}+\frac{u_7-2u_4+u_8}{h^2})=$$
$$[f_0+\frac{h^2}{12}(\frac{f_3-2f_0+f_1}{h^2})+\frac{h^2}{12}(\frac{f_2-2f_0+f_4}{h^2})]+O(h^4),$$

or

$$\frac{1}{12h^2}(12u_3-24u_0+12u_1+u_6-2u_3+u_7-2u_2+4u_0-2u_4+u_5-2u_1+u_8$$
$$+12u_2-24u_0+12u_4+u_6-2u_2+u_5-2u_3+4u_0-2u_1+u_7-2u_4+u_8)=$$
$$\frac{1}{12}(12f_0+f_3-2f_0+f_1+f_2-2f_0+f_4)+O(h^4),$$

or

$$\frac{1}{12h^2}(-40u_0+8u_1+8u_2+8u_3+8u_4+2u_5+2u_6+2u_7+2u_8)=$$
$$\frac{1}{12}(8f_0+f_1+f_2+f_3+f_4)+O(h^4),$$

or

$$\frac{1}{6h^2}\big(-20u_0+4(u_1+u_2+u_3+u_4)+u_5+u_6+u_7+u_8\big)=$$
$$\frac{1}{12}(8f_0+f_1+f_2+f_3+f_4)+O(h^4),$$

or

$$\frac{1}{6h^2}\big(-20u_0+4\sum_{j=1}^{4}u_j+\sum_{j=5}^{8}u_j\big)=\frac{1}{12}\big(8f_0+\sum_{j=1}^{4}f_j\big)+O(h^4) \tag{3.36}$$

Using two index notation

$$u_0=u_{ij}\ ,\quad u_1=u_{i+1,j}\ ,\quad u_2=u_{i,j+1}\ ,\quad u_3=u_{i-1,j}\quad \dots,$$

we rewrite equation (3.36) in the form

$$\begin{aligned}&-20u_{ij}+4(u_{i+1,j}+u_{i,j+1}+u_{i-1,j}+u_{i,j-1})\\&+u_{i+1,j+1}+u_{i-1,j+1}+u_{i-1,j-1}+u_{i+1,j-1}\\&=\frac{h^2}{2}(8f_{ij}+f_{i+1,j}+f_{i,j+1}+f_{i-1,j}+f_{i,j-1}+O(h^4).\end{aligned}\tag{3.37}$$

The fourth-order compact scheme (3.36) or (3.37) is solved either by SOR, by Gauss-Seidel, or by Jacobi iterative methods (Appendix B contains description of these method).

The Jacobi iterative method is given in the form

$$\begin{aligned}u_{ij}^{(k+1)} &= \frac{1}{20}\Bigg(4\big(u_{i+1,j}^{(k)}+u_{i,j+1}^{(k)}+u_{i-1,j}^{(k)}+u_{i,j-1}^{(k)}\big)\\&+\big(u_{i+1,j+1}^{(k)}+u_{i-1,j+1}^{(k)}+u_{i-1,j-1}^{(k)}+u_{i+1,j-1}^{(k)}\big)\\&-\frac{h^2}{2}\big(8f_{ij}^{(k)}+f_{i+1,j}^{(k)}+f_{i,j+1}^{(k)}+f_{i-1,j}^{(k)}+f_{i,j-1}^{(k)}\big)\Bigg).\end{aligned}$$

The Gauss-Seidel iterative method is given in the form

$$\begin{aligned}u_{ij}^{(k+1)} &= \frac{1}{20}\Bigg(4\big(u_{i+1,j}^{(k)}+u_{i,j+1}^{(k)}+u_{i-1,j}^{(k+1)}+u_{i,j-1}^{(k+1)}\big)\\&+\big(u_{i+1,j+1}^{(k)}+u_{i-1,j+1}^{(k)}+u_{i-1,j-1}^{(k+1)}+u_{i+1,j-1}^{(k+1)}\big)\\&-\frac{h^2}{2}\big(8f_{ij}^{(k)}+f_{i+1,j}^{(k)}+f_{i,j+1}^{(k)}+f_{i-1,j}^{(k)}+f_{i,j-1}^{(k)}\big)\Bigg).\end{aligned}$$

The SOR iterative method is given in the form

$$u_{ij}^{(k+1)} = \omega\big(u_{ij}^{(k+1)}\big)_{GS} + (1-\omega)u_{ij}^{(k)},$$

where $\big(u_{ij}^{(k+1)}\big)_{GS}$ is the solution by the Gauss -Seidel method and ω is an over relaxation factor, f_{ij} denotes the approximation to $f(x_i, y_j)$, u_{ij} denote the approximation to $u(x_i, y_j)$, the superscripts (k) and $(k+1)$ denotes the number of iterates. With the initial guess values (zero-th iterates) for the unknowns, the next iterates (first iterates) for the unknowns can be obtained by using these iterative expressions successively. The latest iterates, denoted by the superscript $(k+1)$, are always used in the iteration whenever available.

3.4 Numerical Results for Model Problem

In this section we obtain the numerical solution of boundary value problem (3.21) and (3.22) using the fourth-order compact scheme (3.36), and standard second-order Finite Difference Schemes (3.26). The test problem used is chosen such that the analytical solution is available, so a rigorous comparison can be made.

3.4.1 Test Problem

To construct a test problem with know solution we specify the function u and f in the form

$$u = e^{x+y}, \quad f = -2e^{x+y}, \tag{3.38}$$

on the unit square. We notice that the above solution is smooth. We consider the test problem with Dirichlet boundary condition, i.e. boundary values of u are given.

3.4.2 Techniques to Estimate Convergence

Let us remind as the main definition and facts on the convergence of FDS. Let us have a BVP (PDE with initial and boundary condition)

$$Lu = f, \tag{3.39}$$

where L is a differential operator acting from Ω to F, $L : \Omega \to F$ and Ω, F are set of sufficiently smooth functions.

Consider the finite difference scheme which corresponds to boundary value problem (3.39)

$$L_h u^h = f^h, \tag{3.40}$$

where L_h is a finite difference operator acting from Ω_h to F_h, $L_h : \Omega_h \to F_h$, Ω_h and F_h are set of grids functions.

Definition (error) Let u^h be a solution of equation (3.40) and u be a solution of equation (3.39). The *error* is a grid function $Z^h = u^h - P_h(u)$, where P_h is projection operator acting from Ω to Ω_h, $P_h : \Omega \to \Omega_h$.

Definition (convergence with order k or accuracy of order k) We will say that solution of FDS *converges* to a solution of the BVP with order k if the norm of error in Ω_h is less than or equal to ch^k,

$$\| Z^h \|_{\Omega_h} = \| u^h - P_h(u) \|_{\Omega_h} \leq ch^k, \tag{3.41}$$

where the constants c and k do not depend on h.

Let $err1$ and $err2$ be errors which correspond to grid systems with $N1 \times N1$ nodes for mesh size h and $N2 \times N2$ nodes for mesh size $\frac{h}{2}$, respectively. The definition of convergence yields

$$err1 = \| u^h - P_h(u) \| \leq ch^k,$$

and

$$err2 = \| u^{\frac{h}{2}} - P_{\frac{h}{2}}(u) \| \leq c(\frac{h}{2})^k.$$

Dividing $err1$ by $err2$ we obtain

$$\frac{err1}{err2} \simeq 2^k.$$

Taking the natural logarithm from both sides, we get an approximate value for the rate of convergence

$$k \simeq \frac{ln(\frac{err1}{err2})}{ln2}. \qquad (3.42)$$

3.4.3 Computer Codes

The test problem is solved by the point-SOR, GAUSS-SEIDEL and JACOBI iteration method. The iterative algorithms require nothing more than the ability to determine Ax for any x. In a computer program, a two-dimensional array v is used to store the approximate values of the unknown function at the grid points. The functions $f(x,y)$ and $g(x,y)$ are assumed given. The boundary values are set by using $g(x,y)$. A set of initial guesses is required to start iterative process. The stopping criterion of the iterative process

$$\max_{1 \leq i,j \leq N-1} |u_{i,j}^{(k+1)} - u_{i,j}^{(k)}| \leq \varepsilon, \qquad (3.43)$$

has been included in the algorithm, where ε is the convergence tolerance. The relaxation factor ω is input as a parameter.

3.4.4 Choice of ε in the Iterative Process

It is not reasonable to stop the iterative process with tolerance less than constant times to the error of approximation of FDS. This mean that the choice of ε in the general solution of equation (3.43) has to be

$$\varepsilon \simeq C\,h^k, \quad C - const,$$

where k is equal 2 for the second-order FDS and 4 for the fourth-order FDS. To find the value of C we performed a series of numerical experiments.

Tables 3.1 and 3.2 show the norm of the absolute error between the exact and approximation solutions and residual depending on the choice of ε for different grid sizes . To estimate the error and residual the infinity norm is used . In particular, Table 3.1 shows the value of ε for the fourth-order HOC. The stopping criterion for the iterative process may be made according the following,

$$\varepsilon \approx h^4 \cdot 10^{-4} \div 10^{-5}.$$

Table 3.2 shows the value of ε for the second-order finite difference scheme. In this case, the stopping criterion for the iterative process may be made according the following,

$$\varepsilon \approx h^2 \cdot 10^{-2} \div 10^{-3}.$$

Similar results were observed if we used the Jacobi and the Gauss-Seidel iterative process to solve the test problem.

Table 3.1: Norm of absolute error and residual for fourth-order SOR iterative method

	h=0.2		h=0.1	
ε	error	residual	error	residual
10^{-4}	.1744E-04	.2248E-03	.3546E-04	.6485E-03
10^{-5}	.2423E-05	.2194E-04	.3475E-05	.9378E-04
10^{-6}	.1999E-05	.4517E-05	.4364E-06	.9446E-05
10^{-7}	.1909E-05	.5006E-06	.1323E-06	.7080E-06
10^{-8}	.1917E-05	.2921E-07	.1197E-06	.8936E-07
10^{-9}	.1917E-05	.4273E-08	.1196E-06	.7814E-08
10^{-10}	.1917E-05	.2775E-09	.1195E-06	.8744E-09
10^{-11}	.1917E-05	.4823E-10	.1195E-06	.8576E-10
10^{-12}	.1917E-05	.3444E-11	.1195E-06	.8252E-11
10^{-13}	.1917E-05	.5647E-12	.1195E-06	.1051E-11

	h=0.05		h=0.025	
ε	error	residual	error	residual
10^{-4}	.7039E-03	.2480E-03	.2430E-02	.2842E-03
10^{-5}	.7443E-04	.2619E-04	.2534E-03	.2535E-04
10^{-6}	.7871E-05	.2772E-05	.2554E-04	.2490E-05
10^{-7}	.7345E-06	.2604E-06	.2567E-05	.2263E-06
10^{-8}	.7227E-07	.2753E-07	.2677E-06	.2211E-07
10^{-9}	.3798E-08	.2589E-08	.2560E-07	.2144E-08
10^{-10}	.6941E-08	.2739E-09	.2215E-08	.2118E-09
10^{-11}	.7417E-08	.2577E-10	.2855E-09	.2116E-10
10^{-12}	.7463E-08	.2738E-11	.4467E-09	.2145E-11
10^{-13}	.7469E-08	.2665E-12	4667E-09	.2444E-12

Table 3.2: Norm of absolute error and residual for second-order SOR iterative method

	h=0.2		h=0.1	
ε	error	residual	error	residual
10^{-2}	.2366E-02	.2812E-02	.8227E-03	.1699E-02
10^{-3}	.1387E-02	.3723E-03	.3433E-03	.4884E-03
10^{-4}	.1380E-02	.1056E-04	.3728E-03	.7669E-04
10^{-5}	.1380E-02	.1056E-04	.3563E-03	.1893E-04
10^{-6}	.1381E-02	.1200E-06	.3549E-03	.8257E-06
10^{-7}	.1381E-02	.1026E-07	.3549E-03	.7303E-07
10^{-8}	.1381E-02	.1026E-07	.3549E-03	.1605E-07
10^{-9}	.1381E-02	.3879E-09	.3549E-03	.7018E-09
10^{-10}	.1381E-02	.9665E-10	.3549E-03	.2733E-09
10^{-11}	.1381E-02	.2812E-11	.3549E-03	.3183E-11
	h=0.05		h=0.025	
ε	error	residual	error	residual
10^{-2}	.3513E-01	.2048E-01	.3867E-02	.6258E-02
10^{-3}	.4218E-02	.3817E-03	.1906E-02	.1669E-02
10^{-4}	.3801E-03	.3750E-04	.9251E-04	.2513E-03
10^{-5}	.7219E-04	.3838E-05	.2282E-04	.4619E-05
10^{-6}	.8743E-04	.3887E-06	.2251E-04	.7685E-06
10^{-7}	.8918E-04	.3932E-07	.2240E-04	.1649E-06
10^{-8}	.8938E-04	.3978E-08	.2249E-04	.2879E-07
10^{-9}	.8941E-04	.4024E-09	.2248E-04	.7750E-09
10^{-10}	.8941E-04	.4071E-10	.2248E-04	.1143E-09
10^{-11}	.8941E-04	.4115E-11	.2248E-04	.9861E-11

Table 3.3: Effect of over relaxation factor on convergence. Fourth-order finite difference scheme

h=0.2 (6 interior points)		h=0.1 (11 interior points)		h=0.05 (21 interior points)		h=0.025 (41 interior points)	
$\epsilon = 10^{-6}$		$\epsilon = 10^{-8}$		$\epsilon = 10^{-10}$		$\epsilon = 10^{-12}$	
ω	Number of iterations	ω	Number of iterations	ω	Number of iterations	ω	Number of iterations
1.10	27	1.10	138	1.10	584	1.10	3021
1.20	22	1.20	115	1.20	550	1.20	2540
1.30	**16**	1.30	95	1.30	461	1.30	2130
1.40	22	1.40	78	1.40	383	1.40	1775
1.50	29	1.50	61	1.50	314	1.50	1464
1.60	42	**1.60**	**57**	1.60	252	1.60	1198
1.70	69	1.70	103	**1.70**	**195**	1.70	942
1.80	168	1.80	356	1.80	490	**1.80**	**718**
ω_{opt} is about 1.3		ω_{opt} is about 1.5		ω_{opt} is about 1.7		ω_{opt} is about 1.8	

3.4.5 Choice of ω in SOR Method

The major difficulty with the over relaxation method is the determination of the best value for the over relaxation factor ω. Unfortunately, there is not a good general method for determining the optimal value for the over relaxation factor ω_{opt} .

The optimal value depends on the size of the system of equations (the number of equations) and the nature of the equations (characteristics such as the strength of the diagonal dominance and the structure of the coefficient matrix). As

Table 3.4: Effect of over relaxation factor on convergence. Second-order finite difference scheme

h=0.2 (6 interior points)		h=0.1 (11 interior points)		h=0.05 (21 interior points)		h=0.025 (41 interior points)	
$\epsilon = 10^{-3}$		$\epsilon = 10^{-4}$		$\epsilon = 10^{-5}$		$\epsilon = 10^{-6}$	
ω	Number of iterations	ω	Number of iterations	ω	Number of iterations	ω	Number of iterations
1.20	13	1.20	60	1.20	265	1.20	1157
1.30	**11**	1.30	49	1.30	219	1.30	953
1.40	12	1.40	39	1.40	177	1.40	774
1.50	15	1.50	28	1.50	140	1.50	615
1.60	21	**1.60**	**27**	1.60	105	1.60	471
1.70	27	1.70	40	**1.70**	**69**	1.70	338
1.80	42	1.80	53	1.80	80	1.80	209
1.90	83	1.90	104	1.90	129	**1.90**	**164**
ω_{opt} is about 1.3		ω_{opt} is about 1.6		ω_{opt} is about 1.7		ω_{opt} is about 1.9	

a general rule, large values of ω_{opt} are associated with larger systems of equations.

If $\omega = 1$, the *SOR* method simplifies to the Gauss-Seidel method. It is easy to show that the SOR fails to converge if ω is outside the interval $(0, 2)$. Though technically the term *underrelaxation* should be used when $0 < \omega < 1$, for convenience the term over relaxation is now used for any value of $\omega \in (0, 2)$. Maximum acceleration is obtained for some optimal value of ω. This optimal value will always lie between 1.0 and 2.0 for the Poisson equations.

In general, it is not possible to compute in advance the value of ω that is optimal with respect to the rate of convergence of the SOR. Even when it is possible to compute the optimal value for ω, the computational expense is usually prohibitive. Frequently, some heuristic or experimental estimate is used. Tables 3.3 and 3.4 demonstrate how SOR fourth-order and second-order can speed up the convergence by ω for our test problem for $h = 0.2$ (6 interior points), $h = 0.1$ (11 interior points), $h = 0.05$ (21 interior points) and $h = 0.025$ (41 interior points).

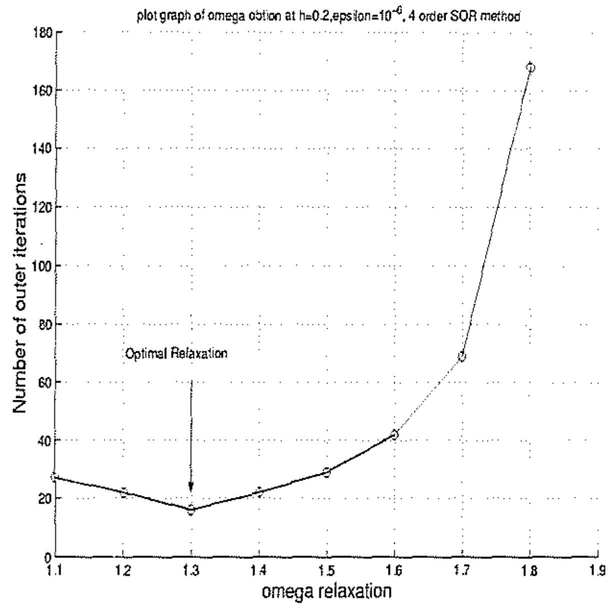

Figure 3.4: The number of iterations in the 4th order SOR method as a function of the relaxation factor ω, $h = 0.2$ and $\varepsilon = 10^{-6}$.

Figure 3.4 - 3.7 demonstrate the dependence of the number of iterations versus relaxation parameter ω, and show this dependence in the case of the fourth-order HOC.

3.4.6 Rate of Convergence

Tables 3.5 - 3.8 show the infinity norm of the absolute errors which are obtained from the grid systems having $N1 \times N1$ nodes. With these values, the resulting rate of convergence is estimated. The rate of convergence is defined

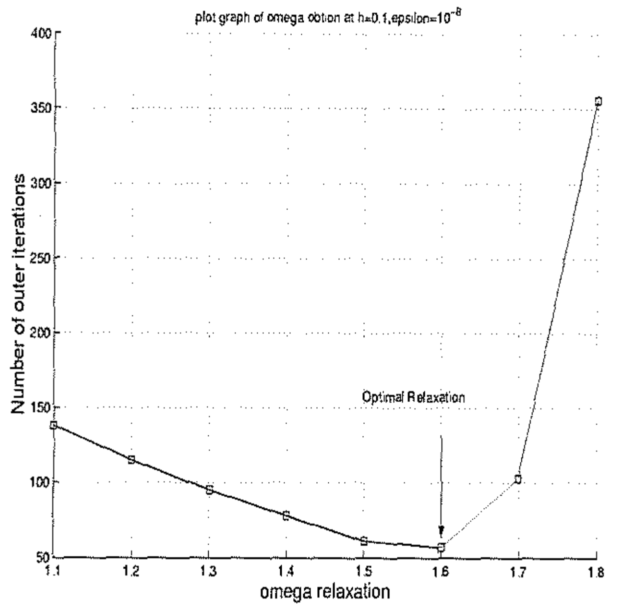

Figure 3.5: The number of iterations in the 4th order SOR method as a function of the relaxation factor ω, $h = 0.1$ and $\varepsilon = 10^{-8}$.

according to equation (3.42)

$$k \simeq \frac{ln(\frac{err_1}{err_2})}{ln2}$$

where $err1$ and $err2$ are errors which correspond to grid systems with $N1 \times N1$ and $N2 \times N2$ nodes, respectively. It is observed that the convergence rate for the h^4 scheme (3.36), is four. This confirms that the compact scheme (3.36) is of fourth-order accuracy when the solutions of (3.19) are smooth.

The error $E = ||u^h - P_h(u)||_\infty$ is graphed in Figure 3.8 against grid size h on log-log scale for the central difference scheme, and fourth-order compact scheme. The rate exponent k (see equation (3.40)) is given by the asymptotic slope of the curves. The slopes of the straight lines on the log-log plot are close to 4 for HOC and close to 2 for CDS.

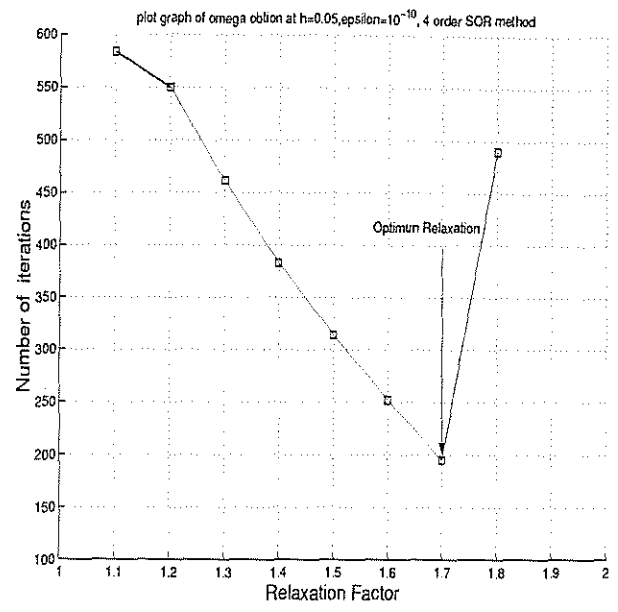

Figure 3.6: The number of iterations in the 4th order SOR method as a function of the relaxation factor ω, $h = 0.05$ and $\varepsilon = 10^{-10}$.

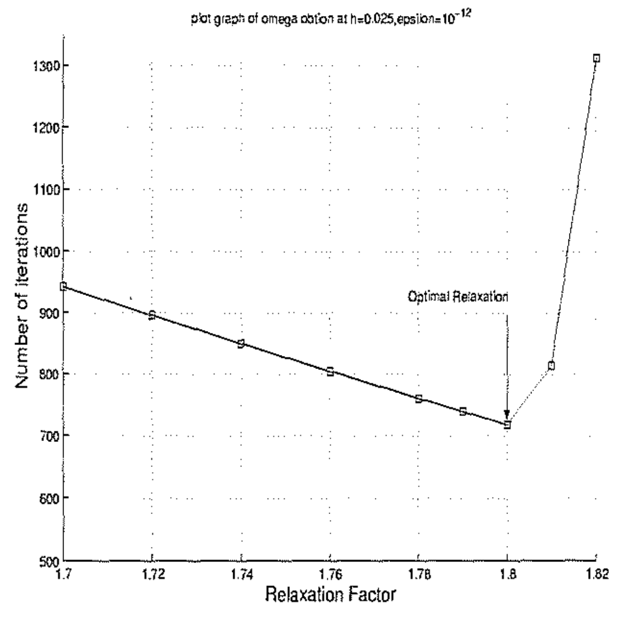

Figure 3.7: The number of iterations in the 4th order SOR method as a function of the relaxation factor ω $h = 0.025$ and $\varepsilon = 10^{-12}$.

Table 3.5: Absolute error and rate of convergence for test problem (3.38), using the Fourth-Oer HOC, SOR iteration method.

h	error $= \|\|u^h - u_{exact}\|\|_\infty$	Rate
0.2	0.1152E-05	-
0.1	0.7229E-07	3.99
0.05	0.5411E-08	3.74
0.025	0.3590E-09	3.91

Table 3.6: Absolute error and rate of convergence for test problem (3.38), using the Fourth-Order HOC, GAUSS-SEIDEL iteration method.

h	error $= \|\|u^h - u_{exact}\|\|_\infty$	Rate
0.2	0.1117E-05	-
0.1	0.6299E-07	4.15
0.05	0.4520E-08	3.80
0.025	0.3384E-09	3.74

Table 3.7: Absolute error and rate of convergence for test problem (3.38), using the Second-Order central difference scheme, SOR iteration method.

h	error $= \|\|u^h - u_{exact}\|\|_\infty$	Rate
0.2	0.7388E-03	-
0.1	0.1662E-03	2.15
0.05	0.4610E-04	1.85
0.025	0.8998E-05	2.36

Table 3.8: Absolute error and rate of convergence for test problem (3.38), using the Second-Order central difference scheme, GAUSS-SEIDEL iteration method.

h	error $= \|\|u^h - u_{exact}\|\|_\infty$	Rate
0.2	0.1325E-02	-
0.1	0.2844E-03	2.22
0.05	0.5700E-04	2.32
0.025	0.1080E-04	2.39

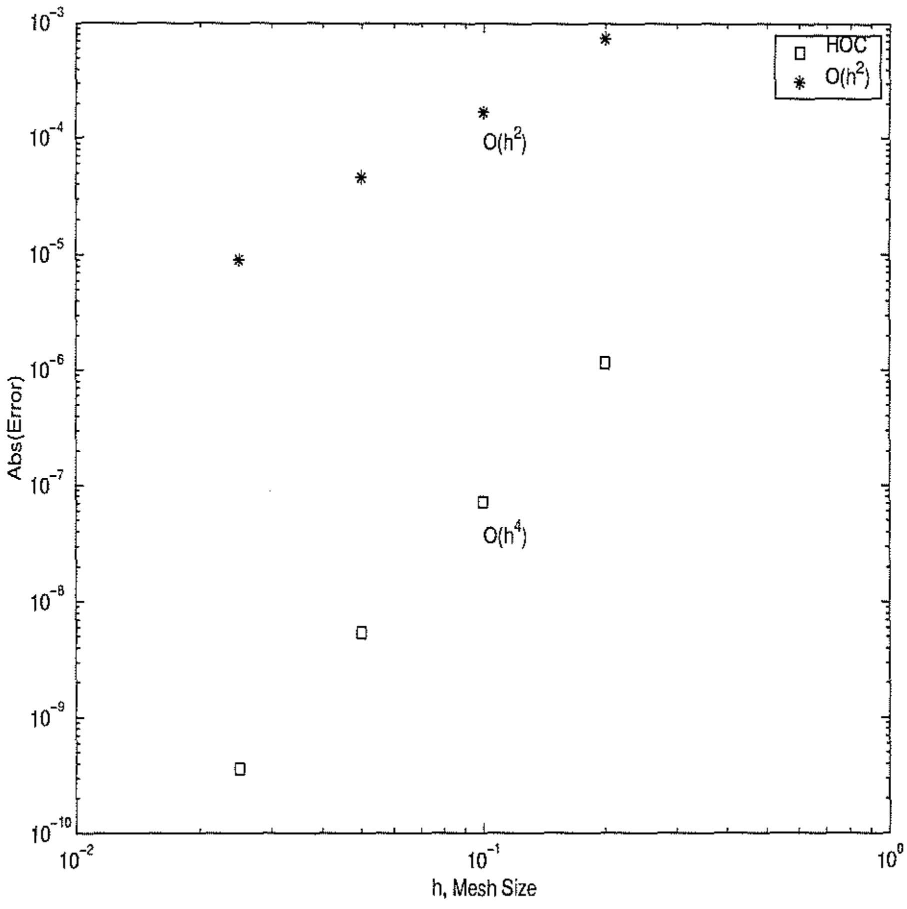

Figure 3.8: Convergence results for the 2D test problem.

Chapter IV

High-Order Compact Finite Difference Scheme for the Euler Equations

4.1 Introduction

In this chapter, we develop the numerical algorithm to solve the boundary value problem in which the governing equations are the steady Euler equations and the vorticity is given on the inflow parts of the boundary. We use the Euler equations in terms of the stream function and vorticity.

4.2 Governing Equations

Let Ω be a bounded domain in $\mathbb{R}^2$ whose boundary $\partial\Omega = \Gamma^1 \cup \Gamma^2 \cup \Gamma^0$, consists of three parts. The parts of the inflow are denoted by Γ^1, the parts of the outflows are denoted by Γ^2, the parts of the impermeable boundary are denoted by Γ^0, the equation of $\partial\Omega$ is given in the following parametric form

$$\begin{aligned} x &= x(s), \\ y &= y(s)\ ,\ s \in [s_0, s_1], \\ x(s_0) &= x(s_1)\ ,\ y(s_0) = y(s_1). \end{aligned} \tag{4.1}$$

The Euler equations governing the fluid flow within domain Ω are

$$\frac{\partial u_1}{\partial t} + u_1\frac{\partial u_1}{\partial x} + u_2\frac{\partial u_1}{\partial y} = -\frac{\partial p}{\partial x}, \tag{4.2}$$

$$\frac{\partial u_2}{\partial t} + u_1 \frac{\partial u_2}{\partial x} + u_2 \frac{\partial u_2}{\partial y} = -\frac{\partial p}{\partial y}, \tag{4.3}$$

$$\frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial y} = 0, \tag{4.4}$$

where $u_i = u_i(x, y, t),\ i = 1, 2$ are the components of the velocity vector, $p = p(x, y, t)$ is the pressure divided by the constant density of the fluid and x, y be are Cartesian coordinates. The boundary conditions are as follow:

$$\begin{aligned}
\Gamma^1 \ &: \ \vec{u}\cdot\vec{n} \ < \ 0, \ \ \vec{u}\cdot\vec{n} = g_1(x(s), y(s)), \\
&\quad \omega = \omega_1(x(s), y(s)), \ (x(s), y(s)) \in \Gamma^1, \\
\Gamma^0 \ &: \ \vec{u}\cdot\vec{n} \ = \ 0, \ (x, y) \in \Gamma^0, \ (x(s), y(s)) \in \Gamma^0, \\
\Gamma^2 \ &: \ \vec{u}\cdot\vec{n} \ > \ 0, \ \ \vec{u}\cdot\vec{n} = g_2(x(s), y(s)), \ (x(s), y(s)) \in \Gamma^2,
\end{aligned} \tag{4.5}$$

where $\vec{n}$ is the outward normal vector to the domain boundary. The correctness of boundary value problem (4.2) - (4.5) was studied by (A. V. Kazhikhov *et al.* (1980)). It is reasonable to reformulate the boundary value problem (4.2) - (4.5) in terms of the vorticity vector ω and stream function ψ. The Euler equations (4.2) - (4.4) for the steady case take the form

$$\psi_{xx} + \psi_{yy} = -\omega, \text{ in } \Omega, \tag{4.6}$$

$$\psi_y \omega_x - \psi_x \omega_y = 0, \text{ in } \Omega, \tag{4.7}$$

where ψ and ω represent the stream function and vorticity vector, respectively (see section 2.5).

The boundary conditions for the stream function and vorticity can be derived from equation (4.5). Let us assume that the boundaries Γ^1 ,Γ^2, Γ^0 consist

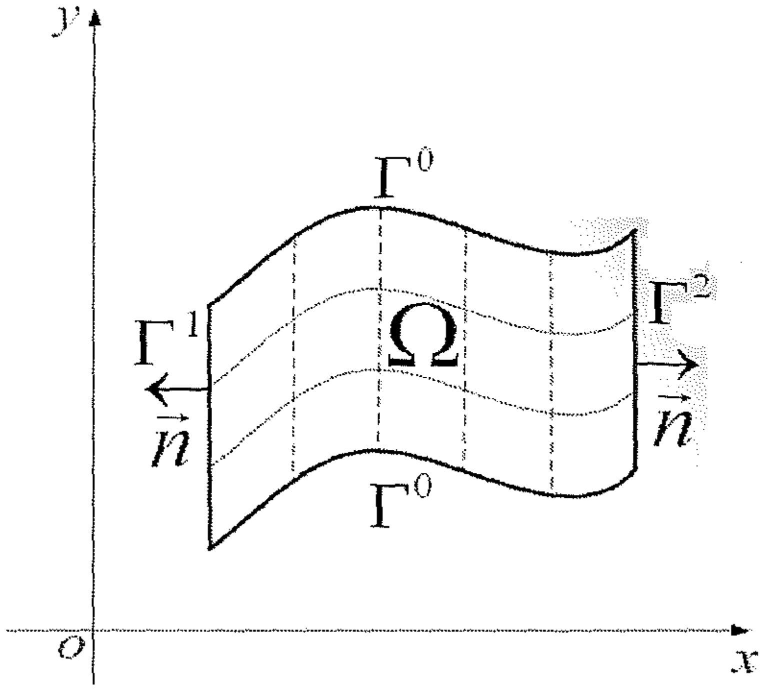

Figure 4.1: Sketch of domain

of several parts

$$\Gamma^1 = \Gamma^1_1 \bigcup \Gamma^1_2 \bigcup \cdots \bigcup \Gamma^1_L,\ s \in [s^1_{11}, s^1_{12}] \bigcup \cdots \bigcup [s^1_{L1}, s^1_{L2}],$$

$$\Gamma^2 = \Gamma^2_1 \bigcup \Gamma^2_2 \bigcup \cdots \bigcup \Gamma^2_K,\ s \in [s^2_{11}, s^2_{12}] \bigcup \cdots \bigcup [s^2_{K1}, s^2_{K2}],$$

$$\Gamma^0 = \Gamma^0_1 \bigcup \Gamma^0_2 \bigcup \cdots \bigcup \Gamma^0_J,\ s \in [s^0_{11}, s^0_{12}] \bigcup \cdots \bigcup [s^0_{J1}, s^0_{J2}],$$

where

$$[s_0, s_1] = \{ \bigcup_{i=1}^{L} [s^1_{i1}, s^1_{i2}] \} \bigcup \{ \bigcup_{j=1}^{K} [s^2_{j1}, s^2_{j2}] \} \bigcup \{ \bigcup_{m=1}^{J} [s^0_{m1}, s^0_{m2}] \}.$$

The boundary conditions for ψ and ω are the following

$$\begin{aligned}
\Gamma^1: \quad & \psi(s) = \psi(s_{l1}^1) + \int_{s_{l1}^1}^{s} g_1(x(s), y(s))ds = \Psi_l(s), \\
& \qquad \omega = \omega_l(s);\ (x(s), y(s)) \in \Gamma^1, \\
& \qquad s \in \left[s_{l1}^1, s_{l2}^1\right],\ l = 1, 2, \cdots, L, \\
\Gamma^2: \quad & \psi(s) = \psi(s_{k1}^2) + \int_{s_{k1}^2}^{s} g_2(x(s), y(s))ds = \Psi_k(s), \\
& \qquad s \in \left[s_{k1}^2, s_{k2}^2\right],\ k = 1, 2, \cdots, K\ ;\ (x(s), y(s)) \in \Gamma^1, \\
\Gamma^0: \quad & \psi(s) = C_{jo};\ s \in \left[s_{j1}^0, s_{j2}^0\right],\ j = 1, 2, \cdots, J.
\end{aligned} \tag{4.8}$$

It is easy to check by direct differentiation that equation (4.7) has a general solution in the form

$$\omega = \widetilde{\omega}(\psi).$$

Here $\widetilde{\omega}(\psi)$ is an arbitrary function of ψ. We have

$$\begin{aligned}
\frac{\partial \omega}{\partial x} &= \frac{\partial \widetilde{\omega}(\psi(x, y))}{\partial x} = \frac{\partial \psi}{\partial x} \cdot \frac{\partial \widetilde{\omega}}{\partial \psi}, \\
\frac{\partial \omega}{\partial y} &= \frac{\partial \widetilde{\omega}(\psi(x, y))}{\partial y} = \frac{\partial \psi}{\partial y} \cdot \frac{\partial \widetilde{\omega}}{\partial \psi}.
\end{aligned}$$

Substituting these formulas into equation (4.7) we get

$$\psi_y \psi_x \widetilde{\omega}_\psi - \psi_y \psi_x \widetilde{\omega}_\psi \equiv 0.$$

In the domain Ω we construct the finite difference grid $\Omega_h = \{(x_i, y_j)\}$. We denote the grid functions at the grid points as ψ_{ij}, ω_{ij}.

Let $\psi_{ij}^{(0)}$ and $\omega_{ij}^{(0)}$ be the initial guesses. To find $\psi_{ij}^{(k)}$ and $\omega_{ij}^{(k)}$ we perform the following steps

1. Solve

$$\Delta \psi_{i,j}^{(k+1)} = -\omega_{ij}^{(k)},\ k = 0, 1, \cdots, K, \tag{4.9}$$

   with boundary condition (4.8) for all inner points of Ω_h.

2. Using the boundary condition on Γ^1, we can find s^*, such that

$$\psi_{ij}^{(k+1)} = \Psi_l(s^*), \tag{4.10}$$

3. Using the boundary condition on Γ^1 for ω

$$\omega_{ij}^{(k+1)} = \omega_l(\bar{s}^*) \tag{4.11}$$

We can stop the iterative process if the following conditions are satisfied,

$$\| \psi^{(\bar{k}+1)} - ij - \psi_{ij}^{(\bar{k})} \| \leq \varepsilon_\psi$$

$$\| \omega_{ij}^{(\bar{k}+1)} - \omega_{ij}^{(\bar{k})} \| \leq \varepsilon_\omega$$

To find the solution of (4.9) we use the results of section 3.2. We substitute $-\omega(\psi)$ for $f(x, y)$.

To find the solution of nonlinear equation (4.10) we use the secant method.

4.3 Numerical Results for the Model Problem

In this section, we obtain the numerical solution for the test problem by using the finite difference scheme presented in section 4.2. A test problem with analytical solution is chosen, and then a rigorous comparison of approximate and exact solutions is performed.

To construct the test problem with known solution, we specify the stream function and vorticity

$$\psi = e^{x+y}, \quad \omega = -2e^{x+y} \tag{4.12}$$

The boundary conditions (see Figure 4.2) to be satisfied are

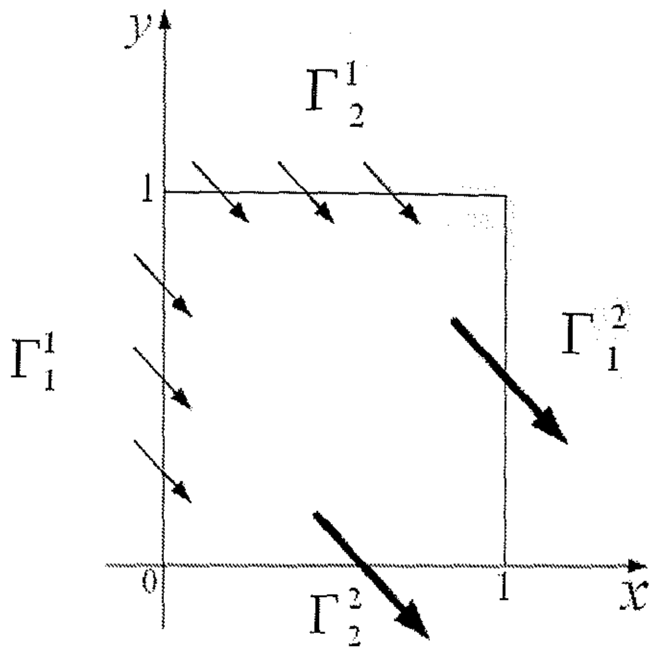

Figure 4.2: Sketch of domain for model problem

$$
\begin{aligned}
\Gamma_1^1 &: \psi(0,y) = f_1(y) = e^y,\ 0 \le y \le 1,\ x = 0,\\
&\quad \omega(0,y) = \xi_1(y) = -2e^y,\ 0 \le y \le 1,\ x = 0,\\
\Gamma_2^1 &: \psi(x,1) = f_2(x) = e^{1+x},\ 0 \le x \le 1,\ y = 1,\\
&\quad \omega(x,1) = \xi_2(x) = -2e^{1+x},\ 0 \le x \le 1,\ y = 1,\\
\Gamma_1^2 &: \psi(1,y) = g_1(y) = e^{1+y},\ 0 \le y \le 1,\ x = 1,\\
\Gamma_2^2 &: \psi(x,0) = g_2(x) = e^x,\ 0 \le x \le 1,\ y = 0.
\end{aligned}
\tag{4.13}
$$

In Figure 4.2 the upper indices correspond to the type of the boundary and the arrows demonstrate the direction of flow. The algorithm developed in section 4.2 is then implemented to these test problems.

These boundary value problems are solved sequentially using a procedure, which is described by the following steps

1. Specify initial values for ω and ψ.

2. Iterate for new ψ values at all points by solving the Poisson equation using

ω from the previous iteration at interior points.

3. Find the point on the boundary Γ_1^1 or Γ_2^1 such that $f_l(s_{ij}^*) = \psi_{ij}$.

4. Find the new ω values at all points by boundary conditions

$$\omega_{ij} = \xi_l(s_{ij}^*)$$

5. Return to step 2 if the solution has not converged yet.

Table 4.1 shows the infinity norm of the absolute errors which are obtained from the grid systems having $N1 \times N1$ nodes. With these values, the resulting rate of convergence is estimated. The rate of convergence is defined by equation (3.42). It is observed that the convergence rate is approximately equal to four. This confirms that the finite difference scheme developed in section 4.2 is of fourth-order accuracy.

Table 4.1: Absolute errors between exact and approximate solutions for stream function and vorticity. Rates of convergence for test problem.

Grid	$err = \|\|\psi^h - \psi_{exact}\|\|_\infty$	Rate	$err = \|\|\omega^h - \omega_{exact}\|\|_\infty$	Rate
6×6	0.173E-05	-	0.345E-05	-
11×11	0.979E-07	4.14	0.196E-06	4.13
21×21	0.579E-08	4.08	0.116E-07	4.08
41×41	0.413E-09	3.81	0.862E-09	3.81

Tables 4.2 and 4.3 show the time which is needed to find the solution of the test problem. In column 2 we show the maximum number of iterations in the interior iterative process to find stream function, column 3 corresponds to total number of iterations to convergence in the last column we show CPU times. We do not need to iterate the Poisson equation for ψ up to convergence. The best

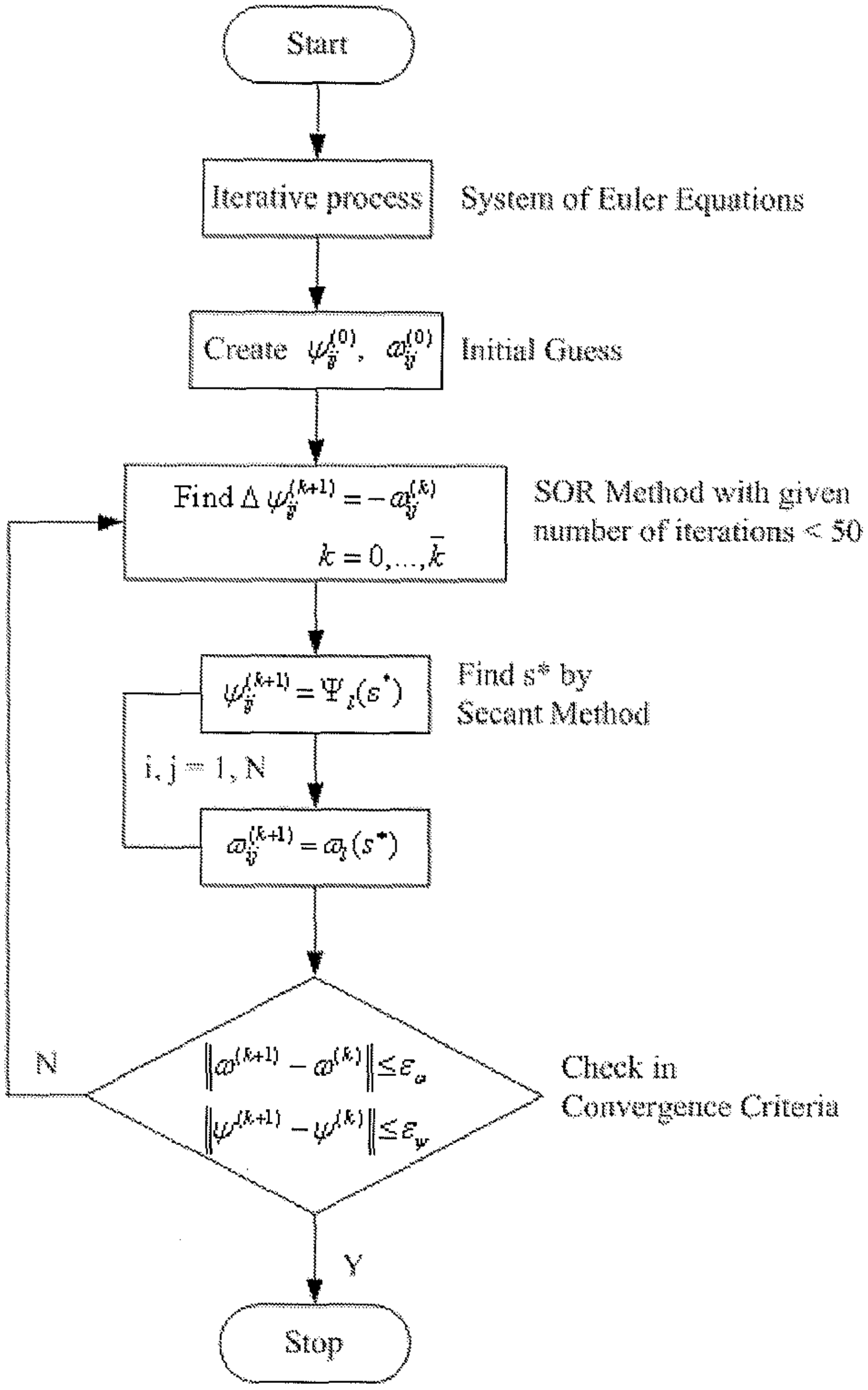
Start
Iterative process
System of Euler Equations
Create $\psi_{ij}^{(0)}$, $\omega_{ij}^{(0)}$
Initial Guess
Find $\Delta\psi_{ij}^{(k+1)} = -\omega_{ij}^{(k)}$
$k = 0, \ldots, \bar{k}$
SOR Method with given number of iterations < 50
$\psi_{ij}^{(k+1)} = \Psi_i(s^*)$
Find s* by Secant Method
i, j = 1, N
$\omega_{ij}^{(k+1)} = \omega_i(s^*)$
$\|\omega^{(k+1)} - \omega^{(k)}\| \le \varepsilon_\omega$
$\|\psi^{(k+1)} - \psi^{(k)}\| \le \varepsilon_\psi$
Check in Convergence Criteria
N
Y
Stop

Figure 4.3: Flow chart of the numerical algorithm.

time corresponds to the case where the number of iterations is in the range of $[50, 100]$ iterations.

The flow of an ideal incompressible fluid corresponding to numerical solu-

Table 4.2: CPU time. Number of iterations for convergence of $2D$ Euler equations. Grid 21×21. $\varepsilon_\psi = \varepsilon_\omega = 10^{-11}$.

sw	Number of Inner Iterations	Nit	Time
1.6	1	228	0.44
	5	47	0.11
	10	24	0.06
	50	11	0.05
	100	12	0.09
	up to convergence but < 10000	12	0.13
1.7	1	176	0.35
	5	36	0.08
	10	18	0.05
	50	12	0.06
	100	12	0.09
	up to convergence but < 10000	12	0.11

tion 4.12 is presented in Figure 4.4 in the form of the isolines $\psi = const$. In the steady case the streamlines coincide with trajectories of fluid particles.

Table 4.3: CPU time. Number of iterations for convergence of $2D$ Euler equations. Grid 21×21. $\varepsilon_\psi = \varepsilon_\omega = 10^{-13}$.

sw	Number of Inner Iterations	Nit	Time
1.7	1	300	2.37
	5	172	1.52
	10	87	0.89
	50	17	0.35
	100	10	0.31
	up to convergence but < 10000	13	1.85
1.8	1	300	2.39
	5	132	1.18
	10	66	0.69
	50	13	0.26
	100	12	0.37
	up to convergence but < 10000	14	1.47

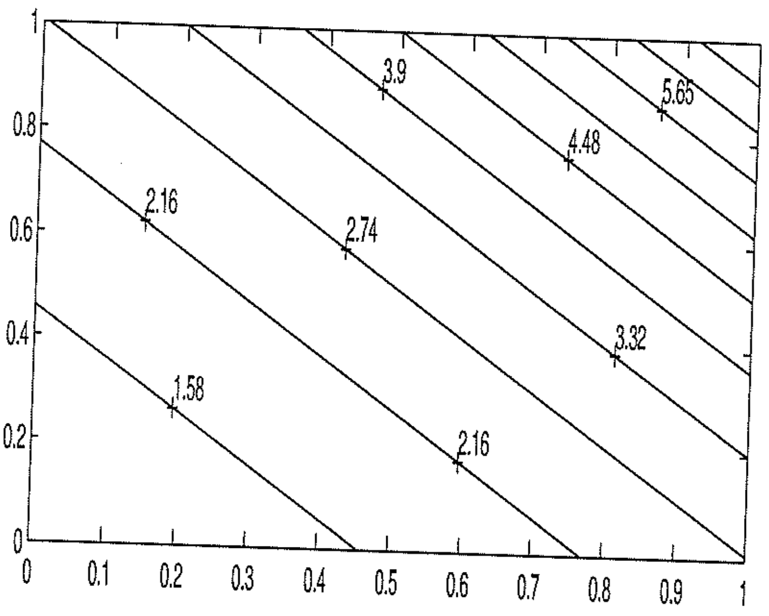

Figure 4.4: Streamlines $\psi = \text{const}$.

Chapter V

Conclusion and Comments on Future Work

The finite difference methods have been the primary computational techniques in CFD. Nowadays, the finite difference methods still play an important role in all branches of CFD. The present thesis provides a set of successful application examples which demonstrate that finite difference methods can be explored to achieve further significant results.

In recent years high-order compact schemes have received considerable attention in the numerical approximation of the partial differential equations. The higher accuracy of the scheme manifests itself as a preferable way to meet the requirement for high resolution in fluid flow regions with large amounts of variation. The compact nature of the schemes effectively reduces the overall work load in solving the nonlinear system resulting from the numerical approximations, since the grid points involved are clustered around the central grid of the computational stencil. In regard to boundary treatment, compact schemes do not require unknown values of fictitious points which are outside the physical domain. This is advantageous over the wide stencil high-order schemes with necessary extrapolations for those fictitious points that in turn brings forth additional problems in regard to accuracy and stability. High-order compact schemes usually have good stability properties.

The fourth-order compact schemes proposed in Chapters III and IV have satisfied these observations. The compact scheme for the steady Euler equations allows decoupled solution procedure for the stream function equation and the vorticity transport equation.

In spite of many existing formulations of fourth-order compact schemes available in the literature, our scheme has the novelty that the coupling relation between the stream function and vorticity (vorticity constant along streamlines) is fully invoked in the derived scheme. Consequently, we have obtained higher efficiency, stability and robustness when applying the scheme to a particular problem. The numerical results in Chapters III and IV supported these conclusions.

The ideas and techniques presented in this thesis are capable of extension and warrant further studies. We envision some aspects which can be considered in future work.

- The underlying strategy of deriving a high-order compact scheme can be readily applied to more general problems of ideal incompressible fluid flow that involve multiply connected domain.
- The high-order compact schemes may be employed with minor adaptation to numerical simulations of ideal incompressible fluid flow within a domain with several inflow and outflow sections.

References

References

Abarbanel, S. and Kumar, A. (1988). "Compact high-order schemes for the Euler equation", **Journal of Scientific Computing** , 1, 275-288.

Antontsev, S. N., Kazhikhov, A. V. and Monakhov V. N. (1990). **Boundary Value Problems in Mechanics of Nonhomogeneous Fluid**, Netherlands.

Atkison, L. V., Harley, P. J. and Hudson, J. D. (1988). **Numerical Methods with Fortran 77**, University of Sheffield.

Batchelor, G. K. (1970). **An Introduction to Fluid Dynamic.** Cambridge: Cambridge University Press.

Chandrasekharaiah D. S. and Debnath L. (1994). **Continuum Mechanics**. Academic Press.

Chorin, A. J. and Marsden J. E. (1997). **A Mathematical Introduction to Fluid Mechanics**. Springer-Verlag, Heidelberg.

Dennis, S. C. R. and Hudson, J. D. (1989). "Compact h^4 finite-difference approximations to operators of Navier-Stokes type", **J. Comput. Phys.**, 85, 390-416.

Dukowicz, J. K. and Ramshaw, J. D. (1979). "Tensor viscosity method for convection in numerical fluid dynamics", **Journal of Computational Psysics**, 33, 71-79.

Fletcher C. A. J. (1991). **Computational Techniques of Fluid Dynamics**. Springer.

Gupta, M. M. (1991). "High accuracy solution of incompressible Navier - Stokes equations", **J. Comput. Phys.**, 93, 343-359.

Hirsh, R. S. (1975). "Higher order accurate difference solution of fluid mechanics problems by a compact differencing technique", **J. Comput. Phys.**, 19, 90-109.

Hoffman, J. D. (1992). **Numerical Methods for Engineers and Scientists.** America: McGraw Hill & Sons.

Kahan, W. (1958). **Gauss-Seidel methods of solving large systems of linear equations**, PhD thesis, University of Toronto.

Kazhikhov, A. V. and Ragulin, V. V. (1980). Nonstationary problems on ideal fluid flow through the bounded domain. **Dokl. Akad. Nayk URSS.** 250 (6).

Lax, P. D. and Wendroff, B. (1964). "Difference schemes for hyperbolic equations with high order of accuracy", **Communications on Pure and Applied Mathematics**, 17, 381-398.

Marchioro, C. and Pulvirenti, M. (1994). **Mathematical Theory of Incompressible Nonviscous Fluid.** Springer-Verlag, New York, Inc.

Mackinnon, R. J. and Carcy, G. F. (1989). "Superconvergence derivatives: A Taylor series analysis", **Int. J. Numer. Meth.in Eng.**, 28, 489-509.

Mackinnon, R. J. and Johnson, R. W. (1991). "Differential equation based representation of truncation errors for accurate numerical simulation", **Int. J. Numer. Meth. in Fluids**, 13, 739-757.

Mackinnon , R. J. and Langerman, M. A. (1991). "A compact high-order finite-different method for elliptic transport problems with variable coefficients", **Int. J. Numer. Meth. in Fluids**, 13, 739-757.

Patterson, J. C. (1983). "General derivative approximations for finite difference schemes", **Int. J. Numer. Meth. in Eng.**, 19, 1235-1241.

Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P. (1992). **Numerical Recipes in FORTRAN**, Cambridge University Press.

Spotz, W. F. and Carey, G. F. (1995). "High-order compact scheme for the stream function vorticity equations", **Int. J. Numer. Methods in Eng.**, 38, 3497-3512.

Warsi, Z. U. A. (1992). **Fluid Dynamics. Theoretical and Computational Approaches**. CRC Press.

Appendix

Appendix A

Iterative Method for the Solution of Nonlinear Equation

A.1 Newton's Method

Consider the sample graph of $y = f(x)$ shown in Figure $A.1$. The root α occurs where the graph crosses the x-axis. We will usually have an estimate of α, and it will be denoted here by x_0. To improve on this estimate, consider the straight line that is tangent to the graph at the point $(x_0, f(x_0))$. If x_0 is near α, this tangent line should be nearly coincident with the graph of $y = f(x)$ for points x about α. Then the root of the tangent line should nearly equal α. This root is denoted here by x_1.

To find a formula for x_1, consider the slope of the tangent line. Using the derivative $f'(x)$, we know from calculus that the slope of the tangent line at $(x_0, f(x_0))$ is $f'(x_0)$. We can also calculate the slope using the fact that the tangent line contains the two points $(x_0, f(x_0))$ and $(x_1, 0)$. This leads to the slope being equal to

$$\frac{f(x_0) - 0}{x_0 - x_1},$$

the difference in the y-coordinates divided by the difference in the x-coordinates. Equating the two different formulas for the slope, we obtain

$$f'(x_0) = \frac{f(x_0)}{x_0 - x_1}.$$

This can be solved to give

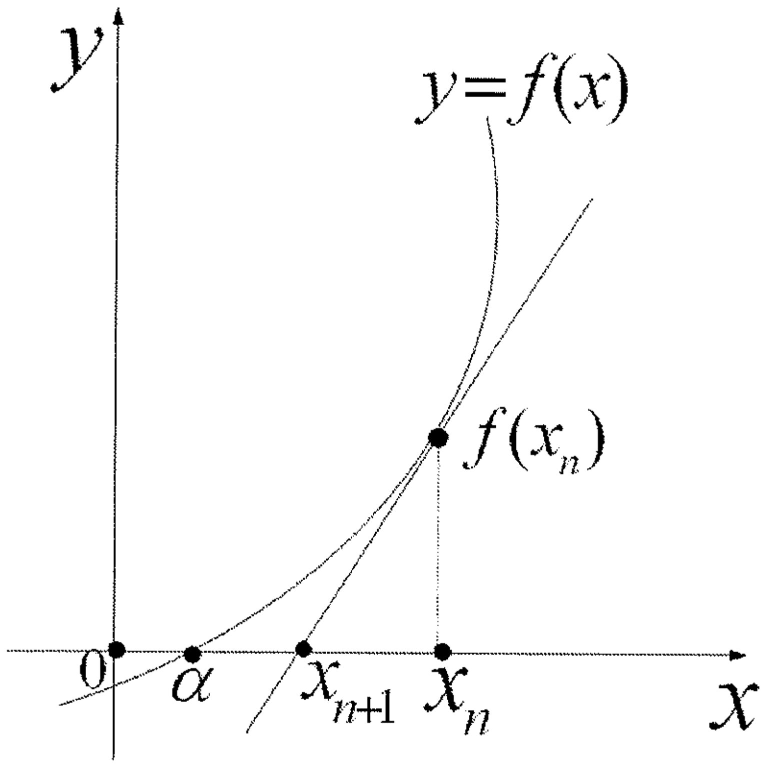

Figure A.1: Geometric interpretation of Newton's method

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}. \tag{A.1}$$

Since x_1 should be and improvement over x_0 as and estimate of α, this entire procedure can be repeated with x_1 as the initial guess. This leads to the new estimate

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}.$$

Repeating this process, we obtain a sequence of numbers $x_1, x_2, x_3, \ldots$ that we hope will approach the root α. These numbers are called iterates, and they are defined recursively by the following general *iteration formula*

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad n = 0, 1, 2 \ldots \tag{A.2}$$

This is *Newton's method* for solving $f(x) = 0$.

A.2 Secant Method

Newton's method is based on approximating the graph $y = f(x)$ with a tangent line and on then using the root of this straight-line as an approximation to the root α of $f(x)$. From this perspective, other straight-line approximations to $y = f(x)$ would also leat to the *secant method.*

Assume that two initial guess to α are known and denote them by x_0 and x_1. They may occur on opposite side of α, as in Figure A.2, or on the same side of α, as in Figure A.3. The two points $(x_0, f(x_0))$ and $(x_1, f(x_1))$, on the graph of $y = f(x)$, determine a straight-line, called a *secant line.* This line is an approximation to the graph of $y = f(x)$, and its root x_2 is an approximation of α.

To derive a formula for x_2, we proceed in a manner similar to that used to derive Newton's method: match the slope determined by $\{(x_1, f(x_1)), (x_2, 0)\}$. This gives

$$\frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{0 - f(x_1)}{x_2 - x_1}.$$

Solving for x_2, we get

$$x_2 = x_1 - f(x_1) \cdot \frac{x_1 - x_0}{f(x_1) - f(x_0)}.$$

Having found x_2, we can drop x_0 and use x_1, x_2 as a new set of approximate values for α. This leads to an improved value x_3; and this process can be continued indefinitely. Doing so, we obtain the general iteration formula

$$x_{n+1} = x_n - f(x_n) \cdot \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}, \quad n \geq 1. \tag{A.3}$$

This is the *secant method* It is called a two point methods, since two approximate values are needed to obtain an improved value.

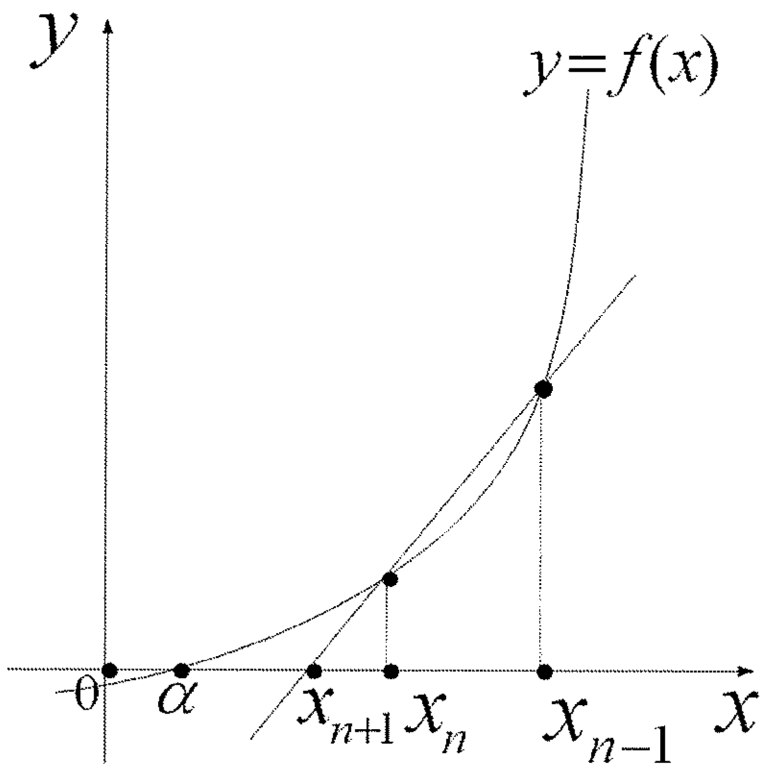

Figure A.2: Geometric interpretation of secant method

Appendix B

Iterative Methods

For many large systems of linear algebraic equations $Ax = b$, the coefficient matrix A is extremely sparse. That is, most of the elements of A are zero. It is generally more efficient to solve such systems of equations by iterative methods than by direct methods. Two iterative methods are presented in this appendix, **Jacobi** iteration and **Gauss-Seidel** iteration.

Iterative methods begin by assuming an initial solution vector $x^{(0)}$. The initial solution vector is used to generate an improved solution vector, $x^{(1)}$, based on some strategy for reducing the difference between $x^{(0)}$ and the actual solution vector x. This procedure is repeated (*iterated*) to convergence. The procedure is *convergent* if each iteration produces approximations to the solution vector that approach the exact solution vector as the number of iterations increases.

Iterative methods do not converge for all sets of equations. Diagonal dominance is a sufficient condition for convergence of the **Jacobi** and the **Gauss-Seidel** methods for any initial solution vector.

A matrix is *diagonal dominant* if the absolute value of each term on the major diagonal is equal to, or larger than, the sum of the values of all the other terms in that row, with the diagonal term being lager than the corresponding sum for at least one row. Thus, diagonal dominance is defined as

$$| a_{ii} | \geq \sum_{j=1, j\neq i}^{n} | a_{ij} | \qquad (i = 1, \dots, n),$$

with $>$ true for at least on row. Some system that are not diagonally dominant may converge for certain initial solution vectors, but convergence is not assured. Iterative methods should generally be avoided for systems of equation that cannot be made diagonally dominant.

When repeated application of an iterative algorithm produces insignificant changes in the solution vector, the procedure should be terminated. In other words, the algorithm is repeated (iterated) until some specified convergence criterion is achieved. Convergence is achieved when some measure of the relative or absolute change in the solution vector is less than a specified convergence criterion. The number of iterations required to achieve convergence depends on

1. the dominance of the diagonal coefficients,
2. the initial solution vector,
3. the algorithm used,
4. the convergence criterion specified.

In general, the stronger the diagonal dominance, the fewer the number of the iterations required to satisfy the convergence criterion.

Jacobi Iteration

Consider the general system of linear algebraic equation $Ax = b$, written in index notation

$$\sum_{j=1}^{n} a_{ij}x_j = b_i \qquad (i = 1, \dots, n).$$

In Jacobi iteration, each equation of the system is solved for the component of the solution vector associated with the diagonal element, that is, x_i. Thus

$$x_i = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j - \sum_{j=i+1}^{n} a_{ij}x_j\right) \qquad (i = 1, \dots, n). \tag{B.1}$$

An initial solution of vector $x^{(0)}$ is chosen. The superscript in parentheses denotes the iteration number, with zero denoting the initial solution vector. The initial solution vector $x^{(0)}$ is substituted in to equation $(B.1)$ to yield the first improved solution vector $x^{(1)}$. Thus,

$$x_i^{(1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(0)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(0)}\right) \qquad (i = 1, \ldots, n). \tag{B.2}$$

This procedure is repeated (iterated) until some convergence criterion is satisfied. The Jacobi algorithm for the general iteration step $(k+1)$ is

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right) \qquad (i = 1, \ldots, n). \tag{B.3}$$

An equivalent, but more convenient, form of equation $(B.3)$ can be obtained by adding and subtracting $x_i^{(k)}$ from equation $(B.3)$ to yield

$$x_i^{(k+1)} = x_i^{(k)} + \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{n} a_{ij}x_j^{(k)}\right) \qquad (i = 1, \ldots, n). \tag{B.4}$$

Equation $(B.4)$ is generally written as

$$x_i^{(k+1)} = x_i^{(k)} + \frac{R_i^{(k)}}{a_{ii}}(i = 1, \ldots, n), \tag{B.5}$$

$$R_i^{(k)} = b_i - \sum_{j=1}^{n} a_{ij}x_j^{(k)}(i = 1, \ldots, n), \tag{B.6}$$

where the term $R_i^{(k)}$ is called the *residual.* The residuals are simply the net values of the equations evaluated for the approximate solution vector $x^{(k)}$.

The Jacobi method is sometimes called the method of *simultaneous* iteration, because all values of x_i are iterated simultaneously. That is, all values of $x_i^{(k+1)}$. The order of processing the equation is immaterial.

The Gauss-Seidel Method

In the Jacobi method, all values of $x_i^{(k+1)}$ are based on $x_i^{(k)}$. The Gauss-Seidel method is similar to the Jacobi method, except that the most recently computed values of all x_i are used in all computations. In brief, as better values of x_i are obtained, use them immediately. Like the Jacobi method, the Gauss-Seidel method requires diagonal dominance to ensure convergence. The Gauss-Seidel algorithm is obtained from the Jacobi algorithm, equation $(B.3)$, by using $x_j^{(k+1)}$ values in the summation from $j = 1$ to $i - 1$ (assuming the sweeps through the equations proceed from $i = 1$ to n). Thus,

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right) \qquad (i = 1, \dots, n). \qquad \text{(B.7)}$$

Equation $(B.7)$ can be written in terms of the residuals R_i by adding and subtracting $x_i^{(k)}$ and rearranging to yield

$$x_i^{(k+1)} = x_i^{(k)} + \frac{R_i^{(k)}}{a_{ii}} \qquad (i = 1, \dots, n), \qquad \text{(B.8)}$$

$$R_i^{(k)} = b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)} \qquad (i = 1, \dots, n). \qquad \text{(B.9)}$$

The Gauss-Seidel method is sometimes called the method of *successive* iteration, because the most recent values of all x_i are used in all the calculations. Gauss-Seidel iteration generally converges faster than Jacobi iteration.

Successive Over Relaxation (SOR)

The Gauss-Seidel method presented above can be modified to include over relaxation simply by multiplying the residual $R_i^{(k)}$ in equation $(B.8)$ by the over relaxation factor ω. Thus, the Successive Over Relaxation method is given by

$$x_i^{(k+1)} = x_i^{(k)} + \omega\frac{R_i^{(k)}}{a_{ii}} \qquad (i = 1, \dots, n), \qquad \text{(B.10)}$$

$$R_i^{(k)} = b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=1}^{n} a_{ij}x_j^{(k)} \qquad (i = 1, \dots, n). \qquad \text{(B.11)}$$

When $\omega = 1$, equation $(B.10)$ yields the Gauss-Seidel method. When $1 < \omega < 2$, the system of equation is *overrelaxed.* Over relaxation is appropriate for systems of linear algebraic equations. When $\omega < 1$, the system of equations is *underrelaxed.* Under relaxation is appropriate when the Gauss-Seidel algorithm causes the solution vector to overshoot, resulting in an oscillatory pattern. This behavior is generally associated with the iterative solution of systems of nonlinear algebraic equations. The iterative method diverges if $\omega \geq 2$. The relaxation factor does not change the final solution, since it multiplies the residual R_i, which is zero when the final solution is reached. The major difficulty with the over relaxation factor ω, is that there is not a good general method for determining the optimum value for the over relaxation factor, ω_{opt}. This optimum value depends on the size of the system of equations (the number of equations) and the nature of the equation (characteristics such as the strength of the diagonal dominance and the structure of the coefficient matrix). As a general rule, larger values of ω_{opt} are associated with larger systems of equations.