

ACCURACY AND EFFICIENCY IMPROVEMENTS IN  
FINITE DIFFERENCE SENSITIVITY CALCULATIONS

A THESIS SUBMITTED TO  
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES  
OF  
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MURAT ÖZHAMAM

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS  
FOR  
THE DEGREE OF MASTER OF SCIENCE  
IN  
AEROSPACE ENGINEERING

DECEMBER 2007

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name :

Signature :

## **ABSTRACT**

### **ACCURACY AND EFFICIENCY IMPROVEMENTS IN FINITE DIFFERENCE SENSITIVITY CALCULATIONS**

Özhamam, Murat

M.S., Department of Aerospace Engineering

Supervisor: Assoc.Prof.Dr. Sinan Eyi

December 2007, 94 Pages

Accuracy of the finite difference sensitivity calculations are improved by calculating the optimum finite difference interval sizes. In an aerodynamic inverse design algorithm, a compressor cascade geometry is perturbed by shape functions and finite differences sensitivity derivatives of the flow variables are calculated with respect to the base geometry flow variables. Sensitivity derivatives are used in an optimization code and a new airfoil is designed verifying given design characteristics. Accurate sensitivities are needed for optimization process. In order to find the optimum finite difference interval size, a method is investigated. Convergence error estimation techniques in iterative solutions and second derivative estimations are investigated to facilitate this method. For validation of the method, analytical sensitivity calculations of Euler equations are used and several applications are performed.

Efficiency of the finite difference sensitivity calculations is improved by parallel computing. Finite difference sensitivity calculations are independent tasks in an inverse aerodynamic design algorithm and can be computed separately. Sensitivity calculations are performed on parallel processors and computing time is decreased.

Keywords: Finite Difference Sensitivity Analysis, Inverse Aerodynamic Design Optimization, Convergence Error Estimation, Parallel Computing

## ÖZ

### **SONLU FARKLARLA DUYARLILIK HESAPLAMALARINDA HASSASİYET VE ETKİNLİĞİN ARTIRILMASI**

Özhamam, Murat

Yüksek Lisans, Havacılık ve Uzay Mühendisliği Bölümü

Tez Yöneticisi: Doç.Dr.Sinan Eyi

Aralık 2007, 94 sayfa

Sonlu farklarla duyarlılık türevi hesaplamalarında, en iyi sonlu fark aralığı hesaplanarak hassasiyet artırılmıştır. Bir ters aerodinamik tasarım algoritması üzerinde, bir kompresör kanatçığı geometrisi şekil fonksiyonlarıyla değiştirilmiş ve temel şekil ile yeni şekil akış verileri arasında sonlu farklar kullanılarak duyarlılık hesapları yapılmıştır. Duyarlılık türevi hesaplamaları bir optimizasyon koduna girilerek istenen özelliklerde yeni bir kanatçık profili tasarlanmıştır. Sonlu farklarla duyarlılık türevi hesaplamaları bu yakınsamada oldukça önemlidir. En iyi sonlu fark miktarını hesaplamak için bir metot üzerine çalışılmıştır. Bu metodun uygulanabilmesi için, yakınsamalı çözüm metotlarındaki yakınsama hatasının hesaplanması ve ikinci derece türev tahminleri üzerinde durulmuştur. Metodun doğrulanması için Euler analitik hassasiyet hesaplamaları kullanılmış ve çeşitli uygulamalar yapılmıştır.

Sonlu farklarla hassasiyet hesaplamalarında etkinlik, paralel işlemciler kullanılarak artırılmıştır. Sonlu farklarla duyarlılık türevi hesaplamaları bağımsız işlemlerdir ve ayrı ayrı hesaplanabilir. Paralel işlemciler kullanılarak hesaplamalar yapılmış ve tasarım süreci kısaltılmıştır.

Anahtar Kelimeler: Sonlu Farklarla Duyarlılık Hesaplamaları, Ters Aerodinamik Tasarım Optimizasyonu, İteratif Hata Tahmini, Paralel İşlem

*To my wife*

*To my parents and sisters*

## **ACKNOWLEDGEMENTS**

The author wishes to thank supervisor Assoc.Prof.Dr. Sinan EYİ for his great advice, criticism and encouragements. This study has been formed by his patience and guidance.

The author would also like to thank 5. Ana Bakım Merkezi Komutanlığı and respectable colleagues for their tolerance in attendance to thesis studies.

The author thanks his family Selahattin, Zübeyde, Elif ÖZHAMAM and Zehra BİROL and his lovely wife Esra ÖZHAMAM for their all good wishes.



## TABLE OF CONTENTS

ABSTRACT .....	iv
ÖZ .....	vi
TABLE OF CONTENTS .....	x
LIST OF TABLES .....	xii
LIST OF FIGURES .....	xiii
LIST OF SYMBOLS .....	xv

## CHAPTERS

1. INTRODUCTION .....	1
1.1 Motivation.....	1
1.2 Objectives .....	4
1.2 Literature Survey.....	5
1.3 Outline .....	6
2. SENSITIVITY ANALYSIS .....	8
2.1 Introduction.....	8
2.2 Flow Analysis .....	8
2.2.1 Euler Equations .....	9
2.2.2 Navier-Stokes Equations .....	9
2.2.3 Euler and Navier-Stokes Equations in Generalized Coordinates.....	12
2.3 Numerical Solution of Euler and Navier Stokes Equations.....	14
2.4 Analytical Sensitivity Analysis.....	15
2.5 Finite Difference Sensitivity Analysis.....	19

2.5.1 Accuracy in Finite Difference Derivatives .....	20
2.5.2 Laplace Solution Example .....	24
3. ACCURACY IMPROVEMENTS IN FD SENSITIVITY CALCULATIONS .....	27
3.1 Introduction.....	27
3.2 Convergence Error Estimation.....	28
3.2.1 Iterative Methods for systems of Equations .....	29
3.2.2 Estimation of Convergence Error in Iterative Solutions .....	31
3.2.3 Application on Laplace's Solution .....	38
3.2.4 Application on Euler and Navier Stokes Solution .....	45
3.3 Optimum Finite Difference Interval estimation.....	54
3.3.1 Case Study I: Laplace Solution .....	58
3.3.2 Case Study II: Euler Solution .....	64
4. EFFICIENCY IMPROVEMENTS IN FD SENSITIVITY ANALYSIS .....	73
4.1 Introduction.....	73
4.2 Parallel Computing.....	73
4.3 Inverse Design Algorithm.....	75
4.4 Case Study .....	76
4.5 Results .....	79
4.6 Discussions .....	85
5. CONCLUSION.....	87
REFERENCES .....	89
APPENDIX .....	92

## LIST OF TABLES

### TABLE

3.1 Sensitivity error, step size and error estimations for Laplaces solution 40x40 ....	60
3.2 Sensitivity error, step size and error estimations for Laplaces solution 80x80 ....	60
4.1 Execution Time and Convergences .....	80
4.2 Parallel Speed-up .....	81

## LIST OF FIGURES

### FIGURE

2.1 Variation of Domain .....	16
2.2 Error variation of $u(x, y, t) = t^4 100xy$ problem w.r.t. differentiation interval h. ....	26
3.1 Variation of Error (Laplace) .....	41
3.2 Variation of Error (Laplace) .....	42
3.3 Variation of Error (Laplace) .....	43
3.4 Variation of Error (Laplace) .....	44
3.5 Solution Meshes.....	45
3.6 Variation of Error (Euler).....	48
3.7 Variation of Error (Euler).....	49
3.8 Variation of Error (Euler).....	50
3.9 Variation of Error (Navier-Stokes) .....	51
3.10 Variation of Error (Navier-Stokes) .....	52
3.11 Variation of Error (Navier-Stokes) .....	53
3.12 Error Variation of Sensitivities (Laplace) .....	61
3.13 Error Variation of Sensitivities (Laplace) .....	62
3.14 Error Variation of Sensitivities (Laplace) .....	63
3.15 Error Variation of Sensitivities (Euler) .....	67
3.16 Error Variation of Sensitivities (Euler) .....	67
3.17 Error Variation of Sensitivities (Euler) .....	68
3.18 Error Variation of Sensitivities (Euler) .....	68
3.19 Pressure distributions of design and target geometries .....	69

3.20 Density distributions of design and target geometries .....	70
3.21 Target and Design geometries .....	71
3.22 Target and Design surface pressure distributions .....	71
3.23 Convergence Parameter variations vs. surface perturbation delta .....	72
3.24 Convergence Parameter variations vs. design cycle .....	72
4.1 Cluster of CPUs .....	75
4.2 Parallel Inverse Design Algorithm.....	76
4.3 257x49 Computational Grid for Rotor R 030 blade .....	78
4.4 Design Practice with 16 Wagner Functions .....	82
4.5 Design Practice with 16 Hick-Henne Functions.....	83
4.6 Design Practice with 16 Patched Polynomials .....	84

## LIST OF SYMBOLS

### ROMAN SYMBOLS

$a$	Speed of sound
$c_p$	Specific heat at constant pressure
CPU	Central processing unit
$D$	Artificial dissipation operator
$e_t$	Total energy per unit volume
$E$	Inviscid flux in x-direction
$E_v$	Viscous flux in x-direction
$\hat{E}$	Inviscid flux in $\xi$ -direction
$\hat{E}_v$	Viscous flux in $\xi$ -direction
$F$	Objective function
$F$	Inviscid flux in y-direction
$F_v$	Viscous flux in y-direction
$\hat{F}$	Inviscid flux in $\eta$ -direction
$\hat{F}_v$	Viscous flux in $\eta$ -direction
$F_{wake}$	Wake parameter
$F_{kleb}$	Klebanoff's intermittency function
$g$	Constraints
$G$	Inviscid flux in z-direction
$G_v$	Viscous flux in z-direction
$\hat{G}$	Inviscid flux in $\zeta$ -direction

$\hat{G}_v$	Viscous flux in $\zeta$ -direction
$J$	Jacobian of the transformation from Cartesian co-ordinate to the generalised curvilinear co-ordinates
$k$	Thermal conductivity
$K$	Clauser parameter
$\ell$	Mixing length
$L$	Reference length
$L$	Spatial discretization operators
$M$	Mach number
$n$	Normal direction
$n$	Problem size
$p$	Pressure
$P$	Number of processors
$Pr$	Prandtl Number
$P, Q$	Forcing functions
$Q$	Vector of dependent flow variables in Cartesian co-ordinate system
$\hat{Q}$	Vector of dependent flow variables in generalized curvilinear co-ordinate system
$q_x, q_y, q_z$	Heat conduction terms in x, y, and z directions
$R$	Riemann invariant
$Re$	Unit Reynolds number
$T$	Execution time
$T$	Temperature
$U_{DIF}$	Difference between maximum and minimum total velocity at a
cons-	stant x-station
$u_\infty$	Free-stream velocity
$u, v, w$	Velocity components in Cartesian co-ordinate system
$U, V, W$	Contravariant velocity components

$y^+$	Non-dimensional wall distance
$x, y, z$	Spatial co-ordinates in Cartesian co-ordinate system
$X$	Design variable

## GREEK SYMBOLS

$\xi, \eta, \zeta$	Components of the generalised curvilinear co-ordinate system
$\partial$	Partial differentiation operator
$\gamma$	Ratio of specific heats
$\lambda$	Coefficient of bulk viscosity
$\rho$	Density
$\mu$	Kinematic viscosity
$\nu_t$	Eddy viscosity
$\tau$	Shear stress
$\tau$	Computational time
$\kappa^{(2)}, \kappa^{(4)}$	Second-and fourth-order dissipation terms
$\nabla$	Total derivatives
$\Omega$	Region of interest
$\Gamma$	Boundary curve
$\omega$	Magnitude of the mean vorticity

## Subscripts

$e$	Extrapolated
$x, y, z, t$	Differentiation with respect to x, y, z, t
$\xi, \eta, \zeta, \tau$	Differentiation with respect to $\xi, \eta, \zeta, \tau$
$\infty$	Freestream property
$max$	Maximum value
$min$	Minimum value



## **Superscripts**

$\wedge$	Transformed
$\sim$	Approximation to the actual value

## **CHAPTER 1**

### **INTRODUCTION**

#### **1.1 Motivation**

The operation of aircraft and propulsion systems relies on performance of their aerodynamic components, such as air intakes, nozzles, wings, cascades, etc. Aerodynamic design of these systems requires engineering expertise, powerful design methods and computer aid. Engineers not only aim to minimize aerodynamic losses and increase performance, but also design the bodies with affordable costs and optimum shapes that can be easily manufactured and maintained. Aerodynamic shape design involves ability to determine the geometry that will satisfy specified aerodynamic objectives. For instance, it is possible to design an airfoil with a specified pressure distribution.

Design procedure is categorized by direct and inverse methods. In direct methods, designer specifies a geometry and then analyzes its performance. Based on the analysis results, the designer modifies the shape in accordance with his experience. This is essentially equivalent to the wind tunnel design method of “trial and error”. Drawback of direct design is that designer should be capable and experienced to reach the required shape.

In inverse design methods, designer is asked to predict the detailed geometry of the flying object so that it is compatible with specified features of the flow field.

Designer usually gets a base geometry and applies some geometry perturbations in order to reach the required flow field or surface parameters such as pressure distribution, lift or pitching moment. Inverse design may lead to unrealistic shapes although solutions are made mathematically correct where upper and lower surfaces can cross-over (fish tail).

In inverse aerodynamic design, flow field design and surface flow design are the main categories. Surface flow design is achieved by specifying a certain flow parameter such as pressure or Mach number on the surface of the body and finding the shape that will generate these surface conditions without regard for the rest of the flow field. The flow field design enforces certain global flow field features such as shock-free flow or minimal entropy generation at every point of the flow field by finding the shape that will satisfy these global constraints. First shape design attempt is done by Joukowski.[1] Joukowski conformal mapping was actually a technique for designing a class of airfoil shapes having specified surface distribution of pressure that corresponds to a flow around a rotating cylinder.

In inverse design, there are two approaches that are used widely: solving an inverse problem and using optimization. Inverse problem computes an airfoil geometry which will produce a desired target pressure distribution without performing any analysis. Some difficulties may arise during the inverse problem. For a given cascade, a flow analysis always produces a pressure distribution at a given flow condition. However, the existence of the solution of the inverse problem is not always guaranteed, i.e. the target pressure distribution may not be generated by any cascade at the given flow condition. Furthermore in inverse design problem, the arbitrary selection of the target pressure distribution may produce a cascade shape that is not physically meaningful.

Some of the difficulties associated with the inverse design problem can be eliminated by using optimization. An optimization-based design method minimizes a specified objective function by combining a flow solver with an optimization

algorithm. Beside the objective function, constraints can be imposed in order to satisfy some performance aspects. In this study, the inverse design is formulated as an unconstrained optimization in which the pressure discrepancy between the target and the designed cascade is the objective function. For a given design condition, an optimization method generally produces a physically meaningful cascade with the appropriate parameterization, even if no cascade will yield the target pressure distribution.

Besides the advantages of optimization over the inverse problem, there are some important issues that have to be considered in order to make this method practical. The computational cost has been a major concern in design optimization. The computational performance of design optimization is greatly influenced by the cost and accuracy of the sensitivity calculations. Choice of flow models, design parameterization, and optimization algorithms also affect the performance of the optimized design.

Another major difficulty in non-linear design optimization is to find the global optimum. Some inverse designs use stochastic methods such as genetic algorithms. Stochastic methods have more advantages in finding globally optimum solution. However, these methods require large number of function evaluations and may not be suitable for practical design applications. In the present study, a deterministic method based on a least-square optimization is used.

The major computational cost in aerodynamic design optimization is associated with the sensitivity calculations that evaluate the variation of the flow field with respect to geometry perturbations. There are two ways to reduce the design computational time. One way is to compute the sensitivities on parallel processors in order to reduce the execution time. The other way is to improve the accuracy of the sensitivity calculations. This also enhances the design performance.

There are two methods to obtain the sensitivities: finite-difference and analytical methods. In analytical method, the sensitivities are obtained by

analytically differentiating the governing equations and the objective and constraint functions with respect to design variables. Analytical sensitivities can be obtained accurately; however, a sensitivity code has to be developed for each analysis code. This causes difficulties especially in complex analysis codes that involve non-differentiable terms.

Unlike the analytical method, the finite-difference method does not require an additional programming effort to build a dedicated sensitivity code. Finite-difference method has two drawbacks. One of them is the computational cost which is associated with the additional flow analyses required for each perturbed geometries. This cost can be reduced by initializing the flow-field of the perturbed geometry from the converged base solution. Parallel computing can also be used to calculate sensitivities on parallel processors in order to reduce the computational cost. The other drawback is the difficulty in predicting the accuracy of the sensitivities. Both of these problems are affected by the size of the finite-difference perturbation and flow variable initialization. However, errors in finite-difference sensitivity calculations can be reduced by developing a method to calculate the sensitivities with optimum perturbation size.

The reliability of a design result also depends on the ability to accurately simulate the flow field. The flow model used in the design process should be able to represent all significant flow physics encountered during the process. In the past, designs were based mostly on the full potential equation. However, the potential formulation can not properly represent the transonic features such as embedded shock waves and shock boundary layer interactions. This study uses Euler and Navier-Stokes solutions that cover shock boundary layer interactions and viscosity.

## **1.2 Objectives**

This study investigates the methods to improve the performances of the finite difference-sensitivity calculations in terms of accuracy and efficiency. In order to improve the accuracy, errors in finite difference sensitivities are to be minimized. An

error analysis was performed to investigate the errors in sensitivity analysis that involve truncation, cancellation and iterative errors. One of the objectives of this study is to estimate these errors accurately. An optimum step size that minimizes error in sensitivity calculation is studied. Calculation of optimum step size requires the estimation of convergence error. Therefore, an extended research has been pursued to estimate the convergence error. The accuracy of finite difference sensitivities are validated with analytical sensitivities. For efficiency improvement, parallel processing is used. Independent finite difference sensitivity calculations are run on different processors in order to decrease the computation time.

### **1.3 Literature Survey**

Iott, Haftka and Adelman [2] already described selecting optimum finite difference interval sizes in sensitivity analyses by finite differences. This study covers structural examples and errors are estimated with the rough error bounds defined. Gill, Murray and Wright [3] covered several algorithms in order to find the optimum step sizes. Barton [4] also describes almost the same method for optimum step size. A recent study is done by Kirsch and Bogomolni [5] covering accuracies of the analytical and finite difference results.

Effects of sensitivity analysis are examined by in both analytical and finite difference approaches. Most of the literature covers the analytical sensitivity derivatives. An adjoint sensitivity analysis is investigated by Chun-ho Sung and Jang Hyuk Kwon [6]. used analytical approaches for aerodynamic design optimization. Both analytical and finite difference methods are compared by Kocabiçak and Eyi [7] in a turbomachinery blade design case and results showed a further study is needed to over the accuracy problem of the finite difference approach.

For calculating the optimum step sizes for finite difference sensitivity derivatives, estimation of the error in the variables are to be done. For estimation of the error in iterative solutions, Ferziger and Peric [8], [9] described a validated method. This method investigates the error propagation by eigenvalue analysis and

successive over relaxation. Successive over relaxation is deeply explained by Hoffmann [10] and Ferziger[11].

Kaplan's thesis [12] covers an inverse design method which couples a Navier-Stokes flow solver and a numerical optimization algorithm. The design method generates a compressor cascade, producing a specified surface pressure distribution at a transonic speed and a least-square optimization technique is used to minimize pressure discrepancies between the target and designed cascades. The same optimization technique and geometry is used in this study. Kocabıçak's thesis [13] covered the effects of sensitivities on the performance of turbomachinery blade design optimization using the Euler equations. Kocabıçak's study performed several inverse design optimizations to evaluate the merits of analytical approach in comparison with the finite-difference approach.

Two valuable AGARD reports [14] [15] includes many good examples of methods for inverse design and optimization. A detailed survey was presented by Sobieczky [16] and Dulikravich [17] to describe the progress in inverse design and optimization. Several remarkable papers by Çetinkaya, Akmandor and Ucer [18] and Vicini and Quagliarella [19] are about shock-free cascade design and usage of genetic algorithms.

PVM software is used for parallel computing and software manual [20] is very detailed and comprehensive. An AGARD report [21] summarizes the techniques with examples.

## **1.4 Outline**

Chapter 2 introduces the basic theory of the flow code for the Euler/Navier-Stokes equations and optimum step size calculation method. Euler and Navier stokes equations are represented in physical and computational domains. Numerical solutions of these equations are presented. Differences and the derivations of analytical and finite difference sensitivity analyses are presented. A brief error

analysis for finite difference sensitivity is done and formulation for optimum finite difference interval size is stated.

In Chapter 3, improvements in accuracy of finite difference sensitivity derivatives are investigated. Values of convergence error and second derivative are used for optimum interval size that minimizes the error. Convergence error estimation method, which is based on eigenvalue analysis, is stated. This method is applied on Laplace, Euler and Navier-Stokes iterative solutions. Second order derivative is estimated by a trial method that uses estimated error values. The optimum interval size method is applied on Laplace and Euler solutions and in both applications, analytical results are used for validation. A design example is performed to emphasize the benefits of improving the accuracy of the finite difference sensitivities.

In chapter 4, efficiency improvements in finite difference sensitivity calculations are investigated by parallel computing. Theory of parallel computing and inverse design optimization is given. Application is done on a Navier-Stokes design solution. Benefits and efficiency of using parallel computing in finite difference sensitivity calculations are discussed.

In chapter 5, the conclusion remarks and recommendations for future studies are presented.



## **CHAPTER 2**

### **SENSITIVITY ANALYSIS**

#### **2.1 Introduction**

Design optimization process requires sensitivities which are the change of objective or constraint function or state variables with respect to design parameters. This study focuses on accurate and efficient calculation of finite-difference sensitivities. In general, state variables are calculated after iterative procedures that involve errors and these errors cause sensitivity inaccuracies. In this chapter, the aerodynamic flow analysis and error propagation in finite difference sensitivity analysis are investigated.

#### **2.2 Flow Analysis**

The flow model should have the capability to retain the flow physics for the given flow conditions. The governing equations of the fluid flow, the employed discretization scheme, the choice of appropriate boundary conditions and grid density are very important factors for better flow simulations. In this study, two-dimensional (2-D) Euler and Navier-Stokes flow analyses are used for calculating analytical and numerical sensitivities.

### 2.2.1 Euler Equations

The universal laws of the conservation of mass, momentum, and energy are the basis of the fundamental equations of fluid dynamics. [22] Euler equations are composed of these conservation equations. The 2-D unsteady compressible Euler equations, in a Cartesian coordinate system, can be written as:

$$\frac{\partial w}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0 \quad (2.1)$$

where

$$w = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}, F = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{pmatrix}, G = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vH \end{pmatrix} \quad (2.2)$$

In these equations,  $w$  is the flow variables vector.  $F$  and  $G$  are the inviscid flux vectors.  $\rho$ ,  $p$ ,  $u$ ,  $v$ ,  $E$ ,  $H$ , are the density, pressure, and velocity components in the  $x$  and  $y$  directions, total energy, total enthalpy respectively. [23] The pressure is obtained from the equation of state:

$$p = \rho R T \quad (2.3)$$

### 2.2.2 Navier-Stokes Equations

The Navier-Stokes equations can be written by adding viscous terms to Euler equations. The 2-D Navier-Stokes equations are a set of four coupled, nonlinear partial differential equations. Upon assuming that body forces and the addition of external heat are negligible, the Navier-Stokes equations can be written in non-dimensional conservation law form as:

$$\frac{\partial w}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = \frac{\partial F_v}{\partial x} + \frac{\partial G_v}{\partial y} \quad (2.4)$$

where  $w$  is the vector of conserved mass defined as:

$$w = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix} \quad (2.5)$$

The inviscid flux vectors,  $F$ , and  $G$  are defined as:

$$F = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{bmatrix}, G = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vH \end{bmatrix} \quad (2.6)$$

where  $\rho$  is the density,  $u$  and  $v$  are the  $x$  and  $y$  components of the velocity vector respectively;  $p$  is pressure,  $E$  is the total energy and  $H$  is the total enthalpy per unit volume. [23]

The viscous stresses are included when Newtonian fluid, where the stress is linearly dependent on the rate of strain, is considered and the viscous flux vectors,  $F_v$ , and  $G_v$  are defined as:

$$F_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \beta_x \end{bmatrix} \quad G_v = \begin{bmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \beta_y \end{bmatrix} \quad (2.7)$$

where

$$\begin{aligned}
\tau_{xx} &= \frac{1}{\text{Re}_\infty} [\lambda(u_x + v_y) + 2\mu u_x] \\
\tau_{yy} &= \frac{1}{\text{Re}_\infty} [\lambda(u_x + v_y) + 2\mu v_x] \\
\tau_{xy} &= \tau_{yx} = \frac{1}{\text{Re}_\infty} \mu(u_y + v_x) \\
\tau_{xz} &= \tau_{zx} = \frac{1}{\text{Re}_\infty} \mu(u_z + w_x) \\
\beta_x &= u\tau_{xx} + v\tau_{yy} - q_x \\
\beta_y &= u\tau_{yx} + v\tau_{yy} - q_y
\end{aligned} \tag{2.8}$$

where  $\lambda$  is bulk and  $\mu$  is dynamic viscosity coefficients. Heat conduction terms are defined as:

$$\begin{aligned}
q_x &= -\frac{\mu}{\text{Re}_\infty \text{Pr}(\gamma-1)M_\infty^2} \frac{\partial T}{\partial x} \\
q_y &= -\frac{\mu}{\text{Re}_\infty \text{Pr}(\gamma-1)M_\infty^2} \frac{\partial T}{\partial y}
\end{aligned} \tag{2.9}$$

The Prandtl number,  $Pr$ , is defined as:

$$Pr = \frac{\mu c_p}{k_\infty} \tag{2.10}$$

where  $c_p$  is the specific heat at constant pressure, and  $k$  is the coefficient of thermal conductivity. The Prandtl number is indicative of the relative ability of the fluid to diffuse momentum and internal energy by molecular mechanisms. The Reynolds number indicates the relative importance of inertial and viscous effects in fluid motion and is shown as:

$$Re = \frac{\rho_\infty u_\infty L}{\mu_\infty} \tag{2.11}$$

Here,  $u_\infty$  is the freestream velocity,  $L$  is the reference length,  $\mu$  is the coefficient of dynamic viscosity, and the subscript  $\infty$  denotes freestream values. For turbulent flows, a turbulence model should be used to specify the coefficients of viscosity and heat conductivity, which appear in the viscous terms in equation (2.8).

### 2.2.3 Euler and Navier-Stokes Equations in Generalized Coordinates

In order to apply the numerical algorithm and boundary conditions easily, the governing equations which are developed in the physical domain or Cartesian coordinates,  $(x, y)$ , should be transformed to the computational domain or generalized coordinates,  $(\xi, \eta)$ . Then Euler equations (2.1) can be written in the transformed domain as:

$$\frac{\partial \hat{w}}{\partial t} + \frac{\partial \hat{F}}{\partial \xi} + \frac{\partial \hat{G}}{\partial \eta} = 0 \quad (2.12)$$

where

$$\hat{w} = J^{-1} \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix} \quad \hat{G} = J^{-1} \begin{bmatrix} \rho V \\ \rho u V + \eta_x p \\ \rho v V + \eta_y p \\ \rho V H \end{bmatrix} \quad \hat{F} = J^{-1} \begin{bmatrix} \rho U \\ \rho u U + \xi_x p \\ \rho v U + \xi_y p \\ \rho U H \end{bmatrix} \quad (2.13)$$

where  $U$  and  $V$  are contravariant velocity components defined as:

$$\begin{aligned} U &= \xi_x u + \xi_y v \\ V &= \eta_x u + \eta_y v \end{aligned} \quad (2.14)$$

where  $\xi_x, \xi_y, \eta_x, \eta_y$  are transformation metrics.

Applying this generalized transformation to the Navier-Stokes equations (2.4),

the following transformed equations are obtained

$$\frac{\partial \hat{W}}{\partial \tau} + \frac{\partial \hat{F}}{\partial \xi} + \frac{\partial \hat{G}}{\partial \eta} = \frac{\partial \hat{F}_v}{\partial \xi} + \frac{\partial \hat{G}_v}{\partial \eta} \quad (2.15)$$

where the inviscid flux terms are:

$$\hat{W} = J^{-1} \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix} \quad \hat{G} = J^{-1} \begin{bmatrix} \rho V \\ \rho u V + \eta_x p \\ \rho v V + \eta_y p \\ \rho V H \end{bmatrix} \quad \hat{F} = J^{-1} \begin{bmatrix} \rho U \\ \rho u U + \xi_x p \\ \rho v U + \xi_y p \\ \rho U H \end{bmatrix} \quad (2.16)$$

while the viscous flux terms are given by:

$$\hat{F}_v = J^{-1} \begin{bmatrix} 0 \\ \xi_x \tau_{xx} + \xi_y \tau_{xy} \\ \xi_x \tau_{yx} + \xi_y \tau_{yy} \\ \xi_x \beta_x + \xi_y \beta_y \end{bmatrix} \quad \hat{G}_v = J^{-1} \begin{bmatrix} 0 \\ \eta_x \tau_{xx} + \eta_y \tau_{xy} \\ \eta_x \tau_{yx} + \eta_y \tau_{yy} \\ \eta_x \beta_x + \eta_y \beta_y \end{bmatrix} \quad (2.17)$$

where

$$\begin{aligned} \tau_{xx} &= \frac{\mu}{\text{Re}_\infty} \left[ (\xi_x u_\xi + \eta_x u_\eta) - (\xi_y v_\xi + \eta_y v_\eta) \right] \\ \tau_{yy} &= \frac{\mu}{\text{Re}_\infty} \left[ (\xi_y v_\xi + \eta_y v_\eta) - (\xi_x u_\xi + \eta_x u_\eta) \right] \\ \tau_{xy} = \tau_{yx} &= \frac{\mu}{\text{Re}_\infty} (\xi_y u_\xi + \eta_y u_\eta + \xi_x v_\xi + \eta_x v_\eta) \end{aligned} \quad (2.18)$$

where the heat conduction terms in the computational space are:

$$\begin{aligned}
q_x &= -\frac{\mu}{\text{Pr Re}_\infty (\gamma-1) M_\infty^2} (\xi_x T_\xi + \eta_x T_\eta) \\
q_x &= -\frac{\mu}{\text{Pr Re}_\infty (\gamma-1) M_\infty^2} (\xi_y T_\xi + \eta_y T_\eta)
\end{aligned} \tag{2.19}$$

### 2.3 Numerical Solution of Euler and Navier-Stokes Equations

It is more eligible to predict complex flow phenomenon with solution of Navier-Stokes equations instead of Euler equations. All of the computations are performed until steady-state conditions are reached. Determination of steady-state condition is obtained by writing the system of equation as:

$$\frac{\partial w}{\partial t} = R(w) \tag{2.20}$$

where  $R$  is the steady-state residual. When norm of residual approaches zero, then  $\partial w / \partial t$  also approaches zero, and the system is said to have reached steady-state. In all calculations, the system is deemed steady-state when norm of residual is reduced by a given order of magnitude from its initial state.

The 2-D compressible Euler and Navier-Stokes equations are solved in conservative form using finite-volume flow code. Flow variables are defined at the cell center, and centered differencing is used for spatial derivatives. Both second and fourth-order artificial viscosity are added for numerical stability.[23] The time integration is performed using an explicit four stage Runge-Kutta scheme. Local time stepping, variable-coefficient implicit residual smoothing, and multigrid methods are implemented to accelerate the convergence to the steady-state solutions. For the cascade flow conditions, inlet, outlet and periodic boundary conditions are used. For Navier-Stokes solutions, a no-slip, adiabatic wall condition is used on the airfoil surface. The Baldwin-Lomax Eddy viscosity model [24] is used for turbulence closure and the transition point is assumed to be located at

fourteen percent of the chord for viscous Navier-Stokes solutions.

## 2.4 Analytical Sensitivity Analysis

In analytical sensitivity calculations, the governing equations and their boundary conditions are differentiated with respect to the design variables. If the governing equations are differentiated after their numerical discretization, the method is called as discrete approach. If the governing equations are differentiated before the numerical discretization, the method is known as a continuum (variational) approach.

In discrete direct differentiation method, the objective function  $f$  is a function of the flow variables vector,  $w$ , and the coordinates of the grid points vector,  $x$ , both of which are functions of the design variable  $X$ .

$$f = f[x(X), w(X)] \quad (2.21)$$

The total rate of change of this function, due to a change in the  $i^{\text{th}}$  component of the design variable,  $X_i$ , is given as:

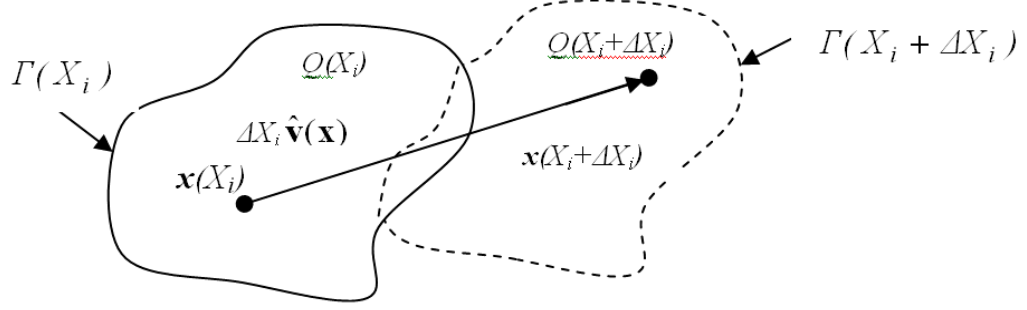
$$\frac{df}{dX_i} = \frac{\partial f}{\partial x} \frac{\partial x}{\partial X_i} + \frac{\partial f}{\partial w} \frac{dw}{dX_i} \quad (2.22)$$

The partial derivatives  $\partial f / \partial x$ ,  $\partial x / \partial X_i$  and  $\partial f / \partial w$  are explicit and they are relatively easy to calculate. However, the response derivative  $d\bar{w} / dX_i$ , is implicitly defined and it is more difficult to evaluate.

The first step in a shape sensitivity analysis is to develop a relationship between a variation in a shape of domain and the resulting variation in the functional defined on the domain. Consider a domain  $Q$ , bounded by the surface  $\Gamma$  and the design variable  $X_i$  as it changes the shape of the body. The changes in the



domain  $Q$  and boundary  $\Gamma$  are shown in Figure 2.1. This approach is called the material derivative approach.



**Figure 2.1** Variation of domain

The derivative with respect to the design variable  $X_i$  is a total derivative,  $dw/dX_i$ , and consists of two parts. The first part will consist of the rate of change of the quantity at a fixed point in space, is often referred to as a local derivative, and is denoted as  $\partial w / \partial X_i$ . The second part will consist of the changes due to variation of the material point's location, and is called convective part,  $\nabla w \cdot \hat{\mathbf{v}}$ , where  $\nabla w$  denotes the spatial gradient of  $w$ , and  $\hat{\mathbf{v}}$  is the design velocity field

$$\hat{\mathbf{v}} = \frac{d\mathbf{x}}{dX_i} \quad (2.23)$$

The total derivative is

$$\frac{dw}{dX_i} = \frac{\partial w}{\partial X_i} + \nabla w \cdot \hat{\mathbf{v}} \quad (2.24)$$

This approach is also called as the material derivative approach, because of the formulation is in the form of a material derivative. The design velocity on surface is

subsequently given as

$$\begin{aligned}\hat{u}_i &= \frac{\partial x}{\partial X_i} \\ \hat{v}_i &= \frac{\partial y}{\partial X_i}\end{aligned}\tag{2.25}$$

To obtain the sensitivity field  $dw/dX_i$ , Euler equations and boundary conditions are differentiated with respect to design variables  $X_i$  as:

$$\frac{d}{dX_i} \left( \frac{\partial w}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} \right) = 0\tag{2.26}$$

Equation (2.26) can be written in the following form:

$$\frac{\partial}{\partial t} \left( \frac{dw}{dX_i} \right) + \frac{\partial}{\partial x} \left( \frac{dF}{dX_i} \right) + \frac{\partial}{\partial y} \left( \frac{dG}{dX_i} \right) - (\nabla F \cdot \frac{\partial \hat{v}}{\partial x}) - (\nabla G \cdot \frac{\partial \hat{v}}{\partial y}) = 0\tag{2.27}$$

The last two terms in Equation (2.37), can be written explicitly as follows:

$$\begin{aligned}& \frac{\partial}{\partial t} \left( \frac{dw}{dX_i} \right) + \frac{\partial}{\partial x} \left( \frac{dF}{dX_i} \right) + \frac{\partial}{\partial y} \left( \frac{dG}{dX_i} \right) \\ & - \frac{\partial F}{\partial x} \frac{\partial \hat{u}}{\partial x} - \frac{\partial F}{\partial y} \frac{\partial \hat{v}}{\partial x} - \frac{\partial G}{\partial x} \frac{\partial \hat{u}}{\partial y} - \frac{\partial G}{\partial y} \frac{\partial \hat{v}}{\partial y} = 0\end{aligned}\tag{2.28}$$

from the Euler equations:

$$\frac{\partial F}{\partial x} = - \left( \frac{\partial w}{\partial t} + \frac{\partial G}{\partial y} \right)\tag{2.29}$$

$$\frac{\partial G}{\partial y} = -\left(\frac{\partial w}{\partial t} + \frac{\partial F}{\partial x}\right) \quad (2.30)$$

Substitution of equation (2.29) and (2.30) into equation (2.28) yields:

$$\begin{aligned} & \frac{\partial}{\partial t} \left( \frac{dw}{dX_i} \right) + \frac{\partial}{\partial x} \left( \frac{dF}{dX_i} \right) + \frac{\partial}{\partial y} \left( \frac{dG}{dX_i} \right) + \left( \frac{\partial w}{\partial t} + \frac{\partial G}{\partial y} \right) \frac{\partial \hat{u}}{\partial x} + \left( \frac{\partial w}{\partial t} + \frac{\partial F}{\partial x} \right) \frac{\partial \hat{v}}{\partial y} \\ & - \frac{\partial f}{\partial y} \frac{\partial \hat{v}}{\partial x} - \frac{\partial g}{\partial x} \frac{\partial \hat{u}}{\partial y} = 0 \end{aligned} \quad (2.31)$$

If we transform Equation (2.31) into generalized coordinates, then the above equation becomes:

$$\begin{aligned} & \frac{\partial}{\partial t} \left( \frac{dw}{dX_i} h \right) + \frac{\partial}{\partial \xi} \left( \frac{dF}{dX_i} \frac{\partial y}{\partial \eta} - \frac{dG}{dX_i} \frac{\partial x}{\partial \eta} \right) + \frac{\partial}{\partial \eta} \left( \frac{dG}{dX_i} \frac{\partial x}{\partial \xi} - \frac{dF}{dX_i} \frac{\partial y}{\partial \xi} \right) \\ & + \frac{\partial}{\partial \xi} \left( F \frac{\partial \hat{v}}{\partial \eta} - G \frac{\partial \hat{u}}{\partial \eta} \right) + \frac{\partial}{\partial \eta} \left( G \frac{\partial \hat{u}}{\partial \xi} - F \frac{\partial \hat{v}}{\partial \xi} \right) \\ & + \frac{\partial w}{\partial t} \left( \frac{\partial \hat{u}}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial \hat{u}}{\partial \eta} \frac{\partial y}{\partial \xi} + \frac{\partial \hat{v}}{\partial \eta} \frac{\partial x}{\partial \xi} - \frac{\partial \hat{v}}{\partial \xi} \frac{\partial x}{\partial \eta} \right) = 0 \end{aligned} \quad (2.32)$$

Equation (2.32) can be written as:

$$\frac{\partial}{\partial t} (\overline{\overline{w}}) + \frac{\partial}{\partial \xi} (\overline{\overline{F}}) + \frac{\partial}{\partial \eta} (\overline{\overline{G}}) + C = 0 \quad (2.33)$$

where

$$\overline{\overline{w}} = \frac{dw}{dX_i} h$$

$$\begin{aligned}
\overline{\overline{F}} &= \frac{dF}{dX_i} \frac{\partial y}{\partial \eta} - \frac{dG}{dX_i} \frac{\partial x}{\partial \eta} \\
\overline{\overline{G}} &= \frac{dG}{dX_i} \frac{\partial x}{\partial \xi} - \frac{dF}{dX_i} \frac{\partial y}{\partial \xi} \\
C &= \frac{\partial}{\partial \xi} \left( F \frac{\partial \hat{v}}{\partial \eta} - G \frac{\partial \hat{u}}{\partial \eta} \right) + \frac{\partial}{\partial \eta} \left( G \frac{\partial \hat{u}}{\partial \xi} - F \frac{\partial \hat{v}}{\partial \xi} \right) + \frac{R_0}{h} \frac{dh}{dX_i}
\end{aligned} \tag{2.34}$$

where  $R_0$  is the steady-state residual in the flow analysis, and can be neglected in the sensitivity calculations, except for the constant term,  $C$ . In this study, the analytical sensitivity solutions in continuum approach are used to validate the accuracy of finite difference sensitivity.

## 2.5 Finite-Difference Sensitivity Analysis

In finite difference approach, sensitivities are calculated from the flow solutions by applying the first order forward finite-difference derivative method. The objective function,  $f[x(X), w(X)]$  is a function of flow variables vector,  $w$ , and grid coordinates vector,  $x$ , which are both functions of design variables,  $X$ . The  $i^{\text{th}}$  component of sensitivity can be approximated using the forward difference calculation, as:

$$\frac{\Delta f}{\Delta X_i} = \frac{f[x(X + \Delta X_i), w(X + \Delta X_i)] - f[x(X), w(X)]}{\Delta X_i} \tag{2.35}$$

Another approach is to first decompose the sensitivities into explicit and implicit parts, and then, apply the finite-difference method. In this approach, the sensitivities of objective function can be written as

$$\frac{df}{dX_i} = \frac{\partial f}{\partial x} \frac{\partial x}{\partial X_i} + \frac{\partial f}{\partial w} \frac{dw}{dX_i} \quad (2.36)$$

where  $dw/dX_i$  is the variation of the flow variables with respect to the design variables. Here,  $\partial f / \partial \bar{x}$ ,  $\partial \bar{x} / \partial X_i$  and  $\partial f / \partial \bar{w}$  are explicit, and can be calculated by either analytical or finite-difference methods accurately. However,  $d\bar{w}/dX_i$  is implicit and to compute it by the finite-difference method requires the solution of multiple iterative flow analyses. This study aims to improve finite-difference sensitivity calculations. The accuracy of the calculations will be investigated below.

### 2.5.1 Accuracy in Finite Difference Derivatives

First order forward difference formula for  $w(X)$  is stated as:

$$\frac{dw}{dX} = \frac{w(X + \Delta X) - w(X)}{\Delta X} \quad (2.37)$$

where  $\Delta X$  is the positive finite difference interval at point  $X$ . There are two dominant errors in the computed value of  $\frac{dw}{dX}$  which are truncation error and cancellation errors. Truncation error is caused by the neglected terms of the Taylor series expansion and cancellation error is caused by the inaccuracies in the computed function values. Taylor series expansion for  $w(X + \Delta X)$  can be represented by analytical values as:

$$\begin{aligned} \tilde{w}(X + \Delta X) = \tilde{w}(X) + \Delta X \frac{d\tilde{w}}{dX} \\ + \frac{\Delta X^2}{2} \frac{d^2 \tilde{w}}{dX^2} + \frac{\Delta X^3}{6} \frac{d^3 \tilde{w}}{dX^3} + \frac{\Delta X^4}{24} \frac{d^4 \tilde{w}}{dX^4} + \dots \end{aligned} \quad (2.38)$$

where  $\tilde{w}$  represents analytical values. Substituting (2.37) into (2.38) as:

$$\begin{aligned} \frac{d\tilde{w}}{dX} &= \frac{\tilde{w}(X + \Delta X) - \tilde{w}(X)}{\Delta X} \\ &\quad - \left( \frac{\Delta X}{2} \frac{d^2 \tilde{w}}{dX^2} + \frac{\Delta X^2}{6} \frac{d^3 \tilde{w}}{dX^3} + \frac{\Delta X^3}{24} \frac{d^4 \tilde{w}}{dX^4} + \dots \right) \end{aligned} \quad (2.40)$$

defining numerical derivative  $\frac{d\tilde{w}}{dX}$  as:

$$\frac{d\tilde{w}}{dX} = \frac{dw}{dX} - \left( \frac{\Delta X}{2} \frac{d^2 \tilde{w}}{dX^2} + \frac{\Delta X^2}{6} \frac{d^3 \tilde{w}}{dX^3} + \frac{\Delta X^3}{24} \frac{d^4 \tilde{w}}{dX^4} + \dots \right) \quad (2.41)$$

The difference between analytical numerical value due to neglected terms of Taylor series expansion is the truncation error and can be written as:

$$\mathcal{E}_{\text{truncation} \frac{dw}{dX}} = \frac{\Delta X}{2} \left| \frac{d^2 w}{dX^2} \right| \quad (2.42)$$

neglecting higher order terms. Cancellation error is due to inaccuracies of the computed values of state variables. Finite difference sensitivities with computed values can be calculated as:

$$\frac{dw}{dX} = \frac{\tilde{w}(X + \Delta X) - \tilde{w}(X)}{\Delta X} \quad (2.43)$$

$$\frac{dw}{dX} = \frac{w(X + \Delta X) - w(X)}{\Delta X} + \frac{\mathcal{E}(X + \Delta X) - \mathcal{E}(X)}{\Delta X} \quad (2.44)$$

where  $\mathcal{E}(X)$  is the error vector that includes round-off and convergence errors. In an iterative procedure, errors  $\mathcal{E}(X + \Delta X)$  and  $\mathcal{E}(X)$  are approximately same.

Therefore, amount of cancellation error can be represented as:

$$\mathcal{E}_{\text{cancellation} \frac{dw}{dX}} = \frac{2|\mathcal{E}(X)|}{\Delta X} \quad (2.45)$$

The error from these two sources is expressed as;

$$\mathcal{E}_{\text{total} \frac{dw}{dX}} = \mathcal{E}_{\text{truncation} \frac{dw}{dX}} + \mathcal{E}_{\text{cancellation} \frac{dw}{dX}} \quad (2.46)$$

$$\mathcal{E}_{\text{total} \frac{dw}{dX}} = \frac{\Delta X}{2} \left| \frac{d^2 w}{dX^2} \right| + \frac{2|\mathcal{E}(X)|}{\Delta X} \quad (2.47)$$

Round-off errors are neglected. When equation (2.47) is differentiated with respect to  $\Delta X$ , minimum value of total error is obtained as:

$$\Delta X_{\text{optimum} \frac{dw}{dX}} = 2 \sqrt{\frac{|\mathcal{E}(X)|}{\left| \frac{d^2 w}{dX^2} \right|}} \quad (2.48)$$

Above equation shows that in order to calculate optimum step size, error term,  $\mathcal{E}$ , and second derivative,  $\frac{d^2 w}{dX^2}$ , should be known or estimated.

In the same manner, optimum step size can be calculated for second order finite difference derivatives. Second order central difference formula can be expressed as:

$$\Phi(X) = \frac{w(X + \Delta X) - 2w(X) + w(X - \Delta X)}{\Delta X^2} \quad (2.49)$$

where  $\Phi(X)$  is used instead of  $\frac{d^2 w}{dX^2}$ . In order to investigate the amount of errors in

this formula Taylor series expansion and cancellation errors are to be expressed:

$$\begin{aligned}\tilde{w}(X + \Delta X) &= \tilde{w}(X) + \Delta X \tilde{w}'(X) \\ &+ \frac{\Delta X^2}{2} \tilde{w}''(X) + \frac{\Delta X^3}{6} \tilde{w}'''(X) + \frac{\Delta X^4}{6} \tilde{w}^{(4)}(X) + \dots\end{aligned}\quad (2.57)$$

$$\begin{aligned}\tilde{w}(X - \Delta X) &= \tilde{w}(X) - \Delta X \tilde{w}'(X) \\ &+ \frac{\Delta X^2}{2} \tilde{w}''(X) - \frac{\Delta X^3}{6} \tilde{w}'''(X) + \frac{\Delta X^4}{6} \tilde{w}^{(4)}(X) + \dots\end{aligned}\quad (2.58)$$

plugging equation (2.57) and (2.58) into equation (2.56), it is obtained that:

$$\frac{\tilde{w}(X + \Delta X) - 2\tilde{w}(X) + \tilde{w}(X - \Delta X)}{\Delta X^2} = \Phi(X) + \frac{\Delta X^2}{12} \tilde{w}^{(4)}(X) + \dots\quad (2.59)$$

Truncation error for  $\Phi(X)$  is the remaining terms on RH as:

$$\mathcal{E}_{truncation\ \Phi} = \frac{\Delta X^2}{12} \tilde{w}^{(4)}(X) + \dots\quad (2.60)$$

In order to get cancellation error, equation (2.56) is to be written by computed values and error terms as:

$$\begin{aligned}\bar{\Phi}(X) &= \frac{\tilde{w}(X + \Delta X) - 2\tilde{w}(X) + \tilde{w}(X - \Delta X)}{\Delta X^2} \\ &+ \frac{\mathcal{E}(X + \Delta X) - 2\mathcal{E}(X) + \mathcal{E}(X - \Delta X)}{\Delta X^2}\end{aligned}\quad (2.61)$$

Amount of cancellation error for  $\Phi(x)$  can be at maximum by bound of errors as:



$$\left| \varepsilon_{cancellation \Phi} \right| = 4 \frac{\varepsilon_A}{\Delta X^2} \quad (2.62)$$

Total error for  $\Phi(x)$  can be written as:

$$\left| \varepsilon_{total \Phi} \right| = \frac{1}{12} \Delta X^2 \left| w^{(4)}(X) \right| + 4 \frac{\varepsilon_A}{\Delta X^2} \quad (2.63)$$

by neglecting higher order terms in truncation error.

Throughout this study all domain solutions are investigated and norm values are got in the calculations by using below definitions:

$$\left| X \right|_1 = \sum_{j=1}^K \left| x_j \right| \quad (2.64)$$

$$\left| X \right|_2 = \sum_{j=1}^K \left| x_j \right|^2 \quad (2.65)$$

$$\left| X \right|_\infty = \max \left| x_j \right| \quad j=1,2,\dots,K \quad (2.66)$$

### 2.5.2 Laplace Solution Example

A case study is done for a simple problem that has minimum errors for optimum finite difference intervals for both first and second order finite difference derivatives. A Laplace solution is performed for function:

$$u(x, y, t) = t^4 100xy \quad (2.67)$$

where  $x$ ,  $y$  and  $t$  are variables. The solution is performed by Laplace solution as:

$$\nabla^2 u(x, y, t) = 0 \quad (2.68)$$

in a 40x40 grid. The error amount for the first-order and second-order derivatives

can be written as by (2.54):

$$|\mathcal{E}_{total\ u^*}| = \frac{h}{2}(12t^2 100xy) + \frac{h^2}{6}(24t 100xy) + \frac{h^3}{24}(24 100xy) + 2\frac{\varepsilon}{h} \quad (2.57)$$

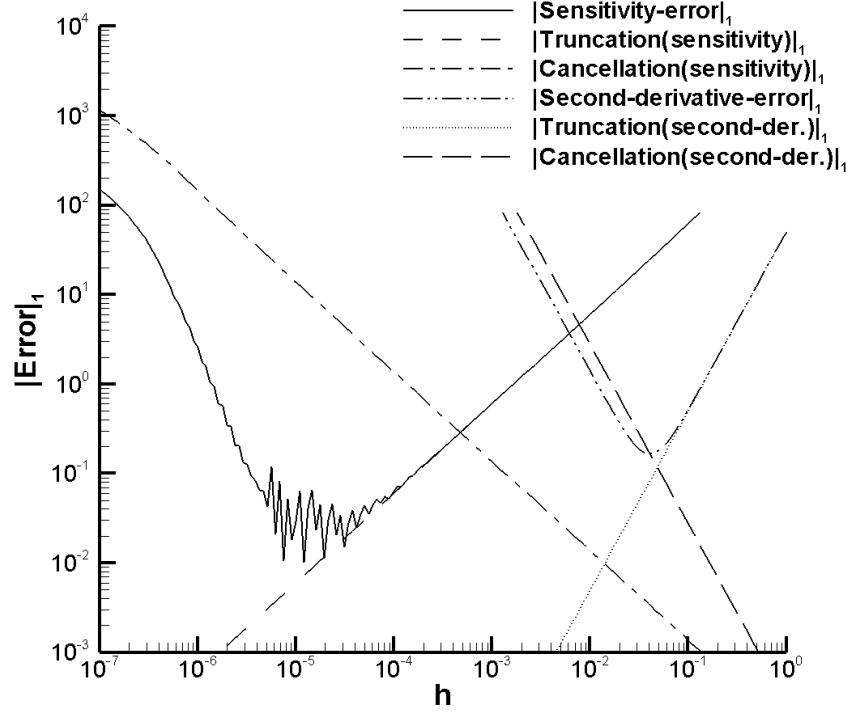
$$|\mathcal{E}_{total\ \Phi}| = \frac{h}{12}(24 100xy) + 4\frac{\varepsilon}{h^2} \quad (2.58)$$

where the  $\varepsilon$  terms are calculated iterative errors and known truncation errors. Error variations due to finite difference intervals for first and second order derivatives in  $L_1$  norm are graphed in Figure 2.2. Example solution performed on 1x1 plate with a 80x80 cartesian grid where the residual reached  $10^{-10}$ .

Exact errors are differences between the analytical formulation and the numerical computation as stated in equation (2.57). Truncation and cancellation errors are generated by the formulations in equation (2.58). Since the function  $u(x, y, t) = t^4 100xy$  can only be differentiated to the 4<sup>th</sup> order, there are no neglected higher order terms for truncation error.

For second order derivative, it can be seen that exact error variation is the sum of the truncation and cancellation errors. Truncation error fits with almost no difference at the larger intervals where the cancellation error is negligible. Cancellation error dominates the total error variation with the smaller intervals. The cancellation error is calculated with equation (2.58) where the error bound is calculated by the iterative error estimation procedure as:

$$|\mathcal{E}_{cancellation\ \Phi}| = 4\frac{|\mathcal{E}_{estimated}|}{h^2} \quad (2.59)$$



**Figure 2.2** Error variation of  $u(x, y, t) = t^4 100xy$  problem w.r.t. differentiation interval  $h$ .

There is a slight difference between the cancellation error and total error due to nominal values in iterative error estimation. In the same manner for the first derivative (sensitivity calculation) truncation error almost is in the same amount of the exact error. Cancellation error is calculated by equation (2.53) by the iterative error estimates as:

$$|\mathcal{E}_{cancellation}| = 2 \frac{|\mathcal{E}_{estimated}|}{h} \quad (2.60)$$

For smaller than the optimum value finite difference intervals, sensitivity error values are oscillatory due to norm calculations. This kind of oscillations is also seen in several studies. [25]

## CHAPTER 3

### ACCURACY IMPROVEMENTS

#### IN FINITE DIFFERENCE SENSITIVITY CALCULATIONS

##### 3.1 Introduction

This study investigates a method for obtaining a near-optimum step size for finite difference sensitivity derivatives. Several studies have already been done for some functions. [2] These methods involve only continuous functions and will be adapted to domain type-solutions in this study.

In finite difference sensitivity calculations, decreasing step size usually decreases truncation error and increases cancellation errors. In chapter 2, the effects of step sizes on the accuracy are investigated and optimum step size for the first-order derivative is stated as:

$$h_{optw'} = 2\sqrt{\frac{|\mathcal{E}|}{|\Phi|}} \quad (3.1)$$

where  $\mathcal{E}$  is the error of the computed value  $f(x)$  and  $\Phi$  is the associated second-order derivative.

In this chapter, methods are investigated for estimating the convergence

error,  $\varepsilon_{conv}$ , and second order derivative,  $\Phi$ , terms in domain solutions, especially for the fluid flow applications. This method will be applied to a Laplace heat problem and an Euler CFD solutions.

### 3.2 Convergence Error Estimation

The result of discretization process is a system of algebraic equations, which are linear or non-linear according to the nature of the partial differential equations. In the non-linear case, the discretized equations can be solved by an iterative technique by guessing an initial solution. Whether the equations are linear or not, efficient methods for solving linear systems of algebraic equations are needed. The system can be written in matrix notation as:

$$[A]w = Q \quad (3.2)$$

where  $A$  is the square sparse coefficient matrix,  $w$  is a vector containing the variable values and  $Q$  is the vector containing the known terms. The matrices derived from partial differential equations are usually sparse.

Direct solution of systems of equations can be considered and well-known methods are Gauss elimination and LU decomposition.[8] Gauss elimination; is the systematic reduction of large systems of equations to smaller ones. By applying row operations some elements are eliminated and a triangular matrix is obtained. Then by back substitution solution is obtained. This method is rather expensive for large systems that are not sparse and interchange of rows by pivoting causes error growth. LU decomposition method factorizes matrix  $A$  into two lower,  $L$ , and upper,  $U$ , triangle matrices and eases the solution as in the Gauss elimination. For factorization Gauss elimination procedures are used. The advantage of LU factorization over Gauss elimination is that the factorization can be performed without knowing vector  $Q$ . This allows that if many systems are involving the same matrix are to be solved, considerable savings can be obtained by performing the factorization first.

Cyclic reduction method is used for sparse matrices which only have non-zero terms on diagonals. For finite difference solutions, these kind of sparse matrices are created and by simple cyclic row operations, solution is reached by back substitutions. Any systems of equations can be solved by Gauss elimination or LU decomposition. Unfortunately, the triangular factors of sparse matrices are not that easy to implement and this hardens the solution algorithm and number of calculations. The discretization error is usually much larger than the accuracy of the computer arithmetic so there is no reason to solve the system that accurately.

### 3.2.1 Iterative Methods for Systems of Equations

Iterative methods are generally used for non-linear systems, an also used for sparse linear systems of equations. In an iterative method, an initial solution is guessed and systematically improved. Generally, iterative methods are more successful than direct methods in number of calculations and usage of memory. [John h. Mathews]

Convergence to the exact solution is the purpose of each iterative solution. Exact solution is the solution that satisfies the equation with zero residual. Considering the simple matrix problem which might result from Finite Difference and Finite Volume approximations of a flow problem as formulated in equation (3.1), after  $n$  iterations,  $w^n$  does not satisfy these equations exactly. Instead there becomes a non-zero residual  $\rho^n$  as;

$$[A]w^n = Q - \rho^n \quad (3.2)$$

where  $w^n$  is the converged solution. Convergence error, which is the difference between exact and iterative solution, can be written as:

$$\mathcal{E}^n = w - w^n \quad (3.3)$$

where  $w$  is the exact solution,  $w^n$  is the converged solution after  $n$  iterations and  $\epsilon^n$  is the associated error vector. The residual can be stated by substituting equation (3.3) into (3.2) as:

$$[A]\epsilon^n = \rho^n \quad (3.4)$$

The purpose of the iteration procedure is to derive the residual to zero. As residual goes to zero, error also tends to be small if convergence is achieved on exact value. Iteration scheme directly affects the convergence and precision of the solution. For iteration scheme example in the solution of equation (3.1), arranging a scheme by decomposing matrix  $A$  as;

$$[A] = [M] - [N] \quad (3.5)$$

and substituting into equation (3.2) as:

$$[M]w^n = [N]w^n + Q - \rho^n \quad (3.6)$$

It can be assumed that as residual,  $\rho^n$ , goes to zero solutions at iterations  $n+1$  and  $n$  are almost equal. Therefore equation (3.6) can be arranged as:

$$[M]w^{n+1} = [N]w^n + Q \quad (3.7)$$

where  $w^{n+1}$  and  $w^n$  are the solutions in referred iterations. Selecting  $M$  and  $N$  matrices plays an important role in solution of matrices. To investigate their effects on error propagation, error definition in equation (3.3) can be used by subtracting  $[M]w^n$  from equation (3.7) as:

$$[M](w^{n+1} - w^n) = Q - ([M] - [N])w^n \quad (3.8)$$

by the definitions in equations (3.2) and (3.5), equation (3.8) can be arranged as:

$$\begin{aligned}
[M](w^{n+1} - w^n) &= Q - [A]w^n \\
[M](w^{n+1} - w^n) &= \rho^n
\end{aligned}
\tag{3.9}$$

By the definition in equation (3.4), equation (3.9) can be stated as:

$$[M](w^{n+1} - w^n) = [A]\epsilon^n \tag{3.10}$$

From equation (3.10), it is understood that convergence error is generated by the iterative solution scheme. For an iterative method to be effective, solution must have minimum number of calculations and converge rapidly. During construction of iterative solution scheme, such as defining  $M$  and  $N$  matrices in the above example, care should be taken to minimize the convergence error growth. Generally  $A$  is a sparse matrix. For convergence,  $M$  should be easily inverted such as being diagonal, tridiagonal, triangular etc.

### 3.2.2 Estimation of Convergence Error in Iterative Solutions

The iterative solution method directly affects the convergence. The convergence of iterative method can be studied by using eigenvalues and eigenvectors of matrix iteration matrices. [8] As residual goes to zero, iterative solution approaches the exact solution. For monotone decreasing iterative solution the eigenvalues of the iteration scheme are to be considered. For the above discussed iterative solution scheme, substituting exact solution into equation (3.7), provided that residual is negligibly small, as:

$$[M]w = [N]w + Q \tag{3.11}$$

and subtracting this from equation (3.7)

$$[M](w^{n+1} - w) = [N](w^n - w) \tag{3.12}$$



and by substituting the convergence error definition in equation (3.3), it obtained as:

$$[M]\varepsilon^{n+1} = [N]\varepsilon^n \quad (3.13)$$

or can be also arranged as:

$$\varepsilon^{n+1} = [M]^{-1}[N]\varepsilon^n \quad (3.14)$$

Equation (3.14) represents that, in an iterative solution, any error on convergence is characterized by the  $[M]^{-1}[N]$  matrix. For convergence,  $\varepsilon^{n+1}$  should be smaller than  $\varepsilon^n$  and converge to minimum value that;

$$\lim_{n \rightarrow \infty} \varepsilon^n = 0 \quad (3.15)$$

Eigenvalues and eigenvectors are important for the convergence of the iteration solution. The eigenvalues and eigenvectors of matrix  $[M]^{-1}[N]$  can be defined as:

$$[M^{-1}N]\psi^k = \lambda_k \psi^k \quad k=1,2,\dots,K \quad (3.16)$$

where  $\psi^k$ 's are the eigenvectors,  $\lambda_k$ 's are the eigenvalues and  $K$  is the number of equations (or grid points if it's a domain solution). Eigenvectors is assumed to be a basis for  $R^n$  vector and forms a complete set. So, initial convergence error can be shown as a combination of eigenvectors such as;

$$\varepsilon^0 = \sum_{k=1}^K a_k \psi^k \quad (3.17)$$

where  $a_k$  is a constant. Iterative solution yields the next iteration error term by combining equation (3.14) and (3.17) as;

$$\mathcal{E}^1 = [\mathbf{M}^{-1} \mathbf{N}] \mathcal{E}^0 = [\mathbf{M}^{-1} \mathbf{N}] \sum_{k=1}^K a_k \psi^k \quad (3.18)$$

and by induction, error for the  $n^{\text{th}}$  iteration can be written as;

$$\mathcal{E}^n = \sum_{k=1}^K a_k \lambda_k \psi^k \quad (3.19)$$

Above equation shows that,  $\lambda_k$ 's should be less than unity for convergence of iteration method. In particular, number of iterations is quite a large number in CFD problems. The largest eigenvalue, also called as the spectral radius, has the dominant effect on error growth. So simply, it will be a reliable assumption that equation (3.20) is to be written by only the spectral radius as;

$$\mathcal{E}^n \approx a_1 (\lambda_1)^n \psi^1 \quad (3.20)$$

where  $\lambda_1$  is the largest eigenvalue,  $a_1$  and  $\psi^1$  are the respective constant and eigenvector.

In an iterative method it is important to be able to estimate the convergence error in order to decide when to stop iteration. Calculation of the eigenvalues of the iteration matrix is difficult that reasonable approximations have to be used. In order to estimate the convergence error some definitions are required. Difference vector between the following iterations is:

$$\delta^n = w^{n+1} - w^n \quad (3.21)$$

By subtracting and adding exact values in equation (3.21), it can be written that;

$$\begin{aligned} \delta^n &= (w^{n+1} - w) - (w^n - w) \\ \delta^n &= \mathcal{E}^{n+1} - \mathcal{E}^n \end{aligned} \quad (3.22)$$

By substituting equation (3.20) into (3.22):

$$\begin{aligned}\delta^n &\approx a_1(\lambda_1)^{n+1}\psi_1 - a_1(\lambda_1)^n\psi_1 \\ \delta^n &\approx (\lambda_1 - 1)a_1(\lambda_1)^n\psi_1\end{aligned}\tag{3.23}$$

In equation (3.23),  $\lambda_1$  and  $a_1$  are scalar quantities where  $\psi^l$  and  $\delta^n$  are vectors. In order to estimate a convergence error, difference term,  $\delta^n$ , is the only computable term. Convergence error is to be written as a combination of  $\delta^n$ . Therefore define a ratio as:

$$\frac{|\delta^n|}{|\delta^{n-1}|} = \frac{(\lambda_1 - 1)a_1(\lambda_1)^n\psi}{(\lambda_1 - 1)a_1(\lambda_1)^{n-1}\psi} = \lambda_1\tag{3.24}$$

where  $|\delta^n|$  represents norm of difference vector  $\delta^n$ . Equation (3.24) shows that largest eigenvalue can be found by the ratio of the differences between two successive iterations. By substituting equation (3.20) into equation (3.23), is obtained as:

$$\delta^n \approx (\lambda_1 - 1)\epsilon^n\tag{3.25}$$

Arranging equation (3.25) by normalized values, convergence error formulation can be obtained as:

$$|\epsilon^n| \approx \frac{|\delta^n|}{\lambda_1 - 1}\tag{3.26}$$

Iterative methods often have complex eigenvalues. When this is the case, the convergence error reduction may not be monotonic. Since the equations are real, complex eigenvalues must occur as conjugate pairs. Their estimation requires an

extension of the above procedure. If the largest eigenvalue is complex equation (3.20) can be written as;

$$\varepsilon^n \approx a_1(\lambda_1)^n \psi_1 + a_1^*(\lambda_1^*)^n \psi_1 \quad (3.27)$$

where  $\lambda_1^*$  is the conjugate pair of  $\lambda_1$  and  $a_1^*$  is the corresponding constant. Residual for iterations in equation (3.27) is to be written for complex values as;

$$\lambda_1 = l e^{iv} \quad (3.28)$$

Ferziger and Peric [8] has stated a formulation to estimate the error growth in an iterative solution, where the largest eigenvalue is complex, by defining a  $z^n$  scalar quantity, which eliminates conjugate terms as;

$$z^n = \delta^{n-2} \delta^n - \delta^{n-1} \delta^{n-1} \quad (3.29)$$

by using  $\delta^n$  vectors in equation (3.25) into (3.36) and defining a parameter  $\sigma$  as;

$$\sigma = (\lambda_1 - 1)a_1 \psi_1 \quad (3.30)$$

Then equation (3.23) for complex problems is to be written as;

$$\delta^n \approx (\lambda_1)^n \omega + (\lambda_1^*)^n \omega^* \quad (3.31)$$

and equation (3.29) gets the form of;

$$z^n = 2l^{2n-2} |\omega|^2 (\cos(2v) - 1) \quad (3.32)$$

and magnitude of the eigenvalue is to be written as;

$$l = \sqrt{\frac{z^n}{z^{n-1}}} \quad (3.40)$$

The complex eigenvalues cause non-monotonic decrease of the convergence error. For estimating convergence error, differences between iterations,  $\delta^n$ , and magnitude of the eigenvalue,  $l$ , are to be computed. Due to complex eigenvalues of and eigenvectors, the result contains terms proportional to the cosine of the phase angle since only magnitudes are interested. It is assumed that these terms are zero in average. This allows a simple relation as:

$$|\mathcal{E}^n| \approx \frac{|\bar{\delta}^n|}{\sqrt{l^2 + 1}} \quad (3.41)$$

This is the desired estimate of the error. Due to oscillations in the solution, this estimate may not be accurate on any particular iteration. Ferziger and Peric (2002) suggests a ratio as;

$$r = \frac{z^n}{|\bar{\delta}^n|^2} \quad (3.42)$$

If the ratio is large the eigenvalue is probably complex. For real eigenvalues  $r$  tends to be smaller than  $10^{-2}$  and for complex eigenvalues  $r$  is approximately 1. Although this method is designed for linear systems, all systems are essentially linear near convergence; so, this method can be applied to non-linear systems as well.

It is recommended that the relation between the largest eigenvalue and residual should also be investigated. [9] Residual of an iterative solution procedure can be changed by successive over relaxation (SOR) methods. [10] In order to accelerate the convergence of iterative procedure, an over-relaxation parameter can be introduced. Residual can be expressed by extrapolating the known residual for the next iteration. Equation (3.2) for iteration  $n+1$ , can be written as:

$$[A]w^{n+1} = Q - \rho^{n+1} \quad (3.43)$$

In a converging solution residual,  $\rho$ , is expected to decrease by successive iterations. It is convenient to represent iterative solution with a extrapolation formulation as:

$$w^{n+1} = w^n + \alpha(w^n - w^{n-1}) \quad (3.44)$$

where  $\alpha$  is an extrapolation parameter. Equation (3.44) can be arranged as:

$$w^{n+1} = (1 + \alpha)w^n - \alpha w^{n-1} \quad (3.45)$$

The formulation can be written as below [11]:

$$w^{n+1} = \omega w^n + (1 - \omega)w^{n-1} \quad (3.46)$$

where

$$\omega = 1 + \alpha \quad (3.47)$$

where  $\omega$  is called the over-relaxation parameter. Ferziger [11] suggests a correlation between the eigenvalues of the iteration matrix and the extrapolation equation (3.46) as:

$$\lambda_w = \omega \lambda + (1 - \omega) \quad (3.48)$$

where  $\lambda_w$  is the over-relaxed solution scheme's eigenvalue. Selecting optimum over relaxation parameter is important that there is an optimum over relaxation factor that corresponds to the eigenvalues of the main iteration matrix. Whenever a larger value than the optimum over relaxation parameter used complex eigenvalues are introduced. [14] Ferziger [11] proposes that in simple Gauss-Seidel iteration scheme, which does use updated solutions in Laplace solution, has absolute eigenvalues of simple Jacobi solution and states that the magnitudes of the largest and minimum

eigenvalues are equal. Where  $\omega$  is between 1 and 2 for over-relaxation,  $\lambda_w$  is minimize when  $\lambda$  is equal to 0. Therefore it can be written that:

$$(1 - \omega) = |\omega\lambda_1^2 + (1 - \omega)| \quad (3.49)$$

$$(1 - \omega) = -\omega\lambda_1^2 - (1 - \omega) \quad (3.50)$$

and optimum relaxation can be found as:

$$\omega_{opt} = \frac{2}{2 - \lambda_1^2} \quad (3.51)$$

It is not simple to find optimum over relaxation factor in larger grids and complicated problems. Generally, larger the grid larger the optimum over relaxation factor is. For values of  $\omega$  less than the optimum, the convergence is monotonic and the rate of convergence increases as  $\omega$  increases. When the optimum  $\omega$  is exceeded, the convergence rate deteriorates and the convergence is oscillatory. [9] For Euler and Navier-Stokes problems, in order to find the best convergence rate and control the eigenvalues, optimum relaxation can be optimized by trial and error. Also this will show whether the eigenvalues are real or not. So error estimation method can also be decided. So, parameters like *CFL* number should be controlled for best convergence.

### 3.2.3 Application on Laplace's Solution

Above mentioned iteration solution method is applied in 2-D Laplace solution as:

$$w_{i,j} = \frac{1}{4}(w_{i+1,j} + w_{i-1,j} + w_{i,j-1} + w_{i,j+1}) \quad (3.52)$$

The linear problem is Laplace's equation in the square domain ( $0 < x < 1$ ,  $0 < y < 1$ ) with boundary conditions of  $100xy$ . The advantage of this solution is that the second

order central difference approximation to the converged solution is exact. The initial guess of the solution is taken as zero everywhere within the domain. A relaxation parameter,  $\omega$ , is introduced in the numerical solution. A FORTRAN code is written for computing and run in single precision of  $10^{-6}$ .

Results of error estimations for uniform grids with 80x80 and 20x20 cells are shown in Figure 3.1 to Figure 3.4 in logarithmic values. Results are represented for  $L_1$ ,  $L_2$  and  $L_\infty$  norms. In each figure, exact convergence error, the convergence error estimate, change between successive iterations and residuals are shown. For each grid size problem, two different relaxation parameters are incorporated: one below the optimum value, where solution has real eigenvalues and one above the optimum leading to complex eigenvalues.

In Figure 3.1 estimation is quite successful for 20x20 grid with a relaxation parameter of 1.5, where the changes between successive iterations are linearly decreasing. Relaxation parameter is smaller than the optimum relaxation parameter and the solution eigenvalues are real. The code itself has already switched to real eigenvalue estimation method with the ratio criteria of 0.1, given by equation (3.42). At the beginning of the solution, when changes and residuals are not monotonic, estimation couldn't be achieved. When the changes approach to computer precision amount, the solution is not monotonic and estimation gives oscillatory results. There is no big difference for estimation quality for different norms.

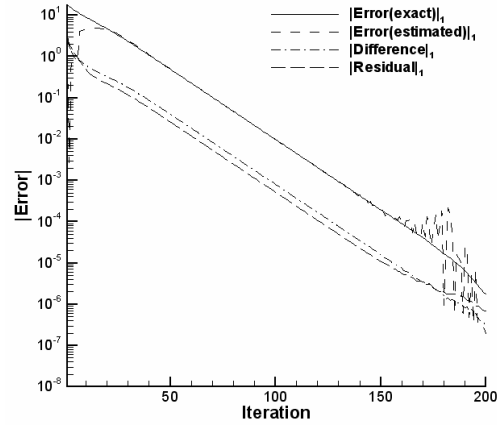
In Figure 3.2 the same solution is done with a relaxation parameter of 1.8 which is greater than the optimum relaxation value. The solution is not monotonic that intends complex eigenvalues. Code has already switched to complex estimation method and estimation is not that successful with respect to the real estimation. Generally a half order of magnitude gap exists between exact error and estimation. There are not much oscillations in this method and solution is reached in almost half number of iterations of the above case where relaxation parameter was smaller.  $L_2$  norm gave the most precise estimation. There are sudden changes at some iterations



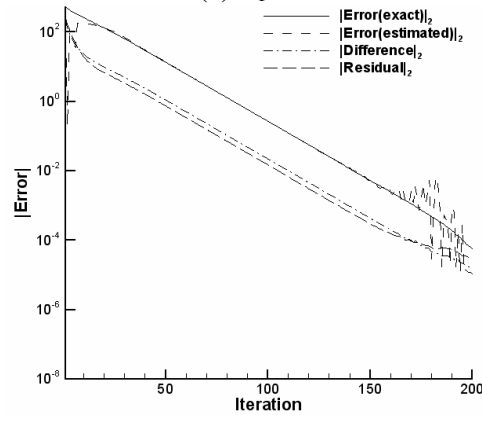
and estimation is linear between these iterations.

In Figure 3.3 80x80 grid solution is represented for a relaxation parameter of 1.5 which is smaller than the optimum parameter. Solution has real eigenvalues and estimation is successful when a monotonic decrease is achieved. In Figure 3.4, 80x80 grid solution is represented for a relaxation parameter of 1.95 which is larger than the optimum parameter. Solution has reached to minimum residual almost 3 times faster than the above case where relaxation parameter was smaller. In this complex estimation, there is again a sudden change in solution and convergence error estimation is again linear. Estimation is not that successful as in the real estimation but again  $L_2$  norm gave the closest estimation.

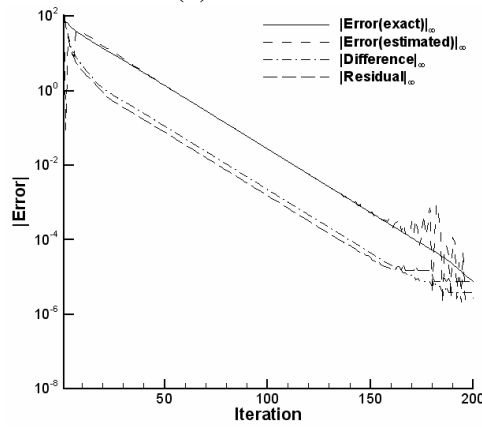
For the case of real eigenvalues, results are smoothly converged. The convergence error estimate is almost exact in this case. However at the beginning of the iterations the norms of differences fall too rapidly and do not follow the fall of the convergence error. This effect is seen well in fine grid. Once the asymptotic reduction rate is achieved, the slopes of all curves are the same. In the case for which the eigenvalues of the iteration matrix are complex, the convergence is not monotonic and there are oscillations in the error. The comparison of predicted and exact errors in this case is also reasonably satisfactory especially in slow solutions with smaller relaxation parameters.



(a)  $L_1$  norm

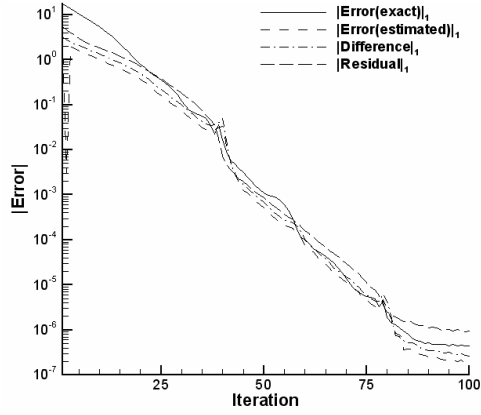


(b)  $L_2$  norm

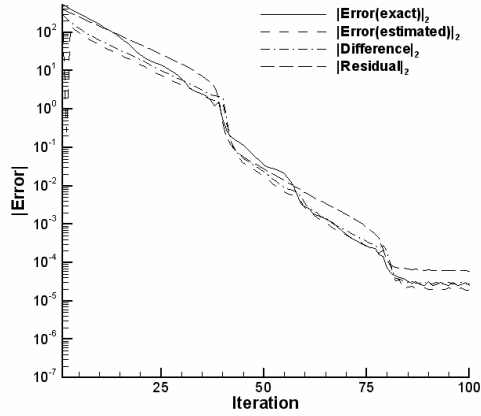


(c)  $L_\infty$  norm

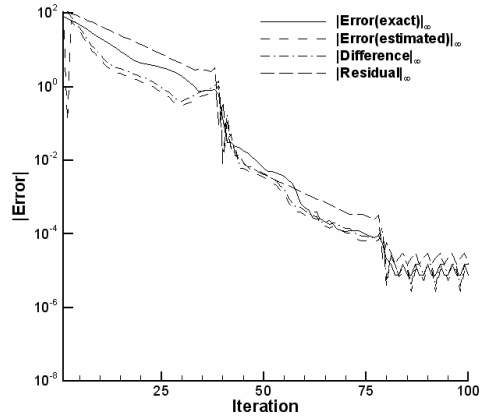
**Figure 3.1** Variation of exact convergence error, error estimate, differences between two successive iterations and residuals for the Laplace problem on a 20x20 Cartesian square grid with a relaxation parameter of 1.5.



(a)  $L_1$  norm

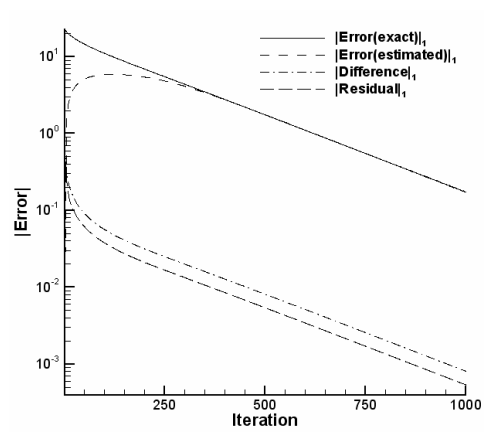


(b)  $L_2$  norm

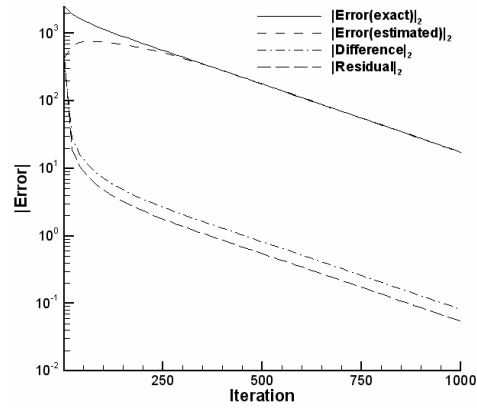


(c)  $L_\infty$  norm

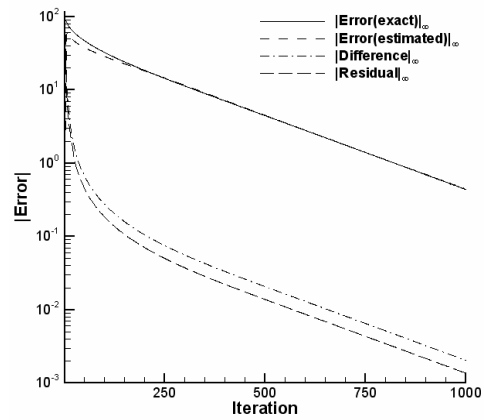
**Figure 3.2** Variation of exact convergence error, error estimate, differences between two successive iterations and residuals for the Laplace problem on a 20x20 Cartesian square grid with a relaxation parameter of 1.80.



(a)  $L_1$  norm

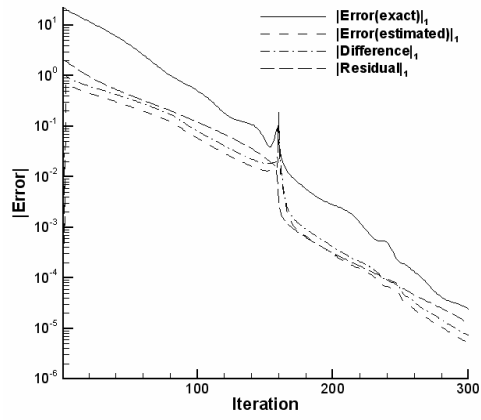


(b)  $L_2$  norm

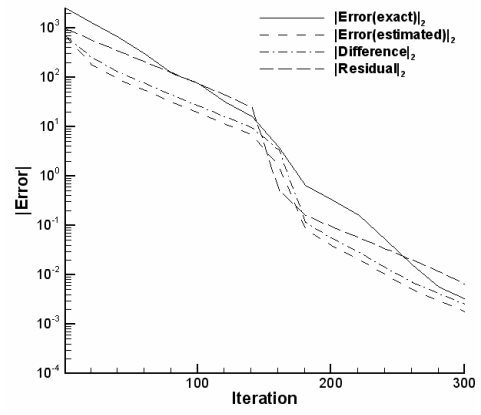


(c)  $L_\infty$  norm

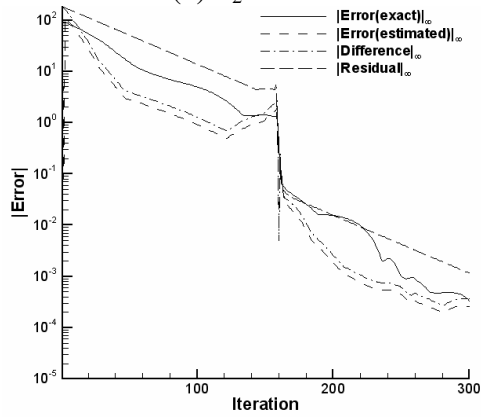
**Figure 3.3** Variation of exact convergence error, error estimate, differences between two successive iterations and residuals for the Laplace problem on a  $80 \times 80$  Cartesian square grid with a relaxation parameter of 1.5.



(a)  $L_1$  norm



(b)  $L_2$  norm



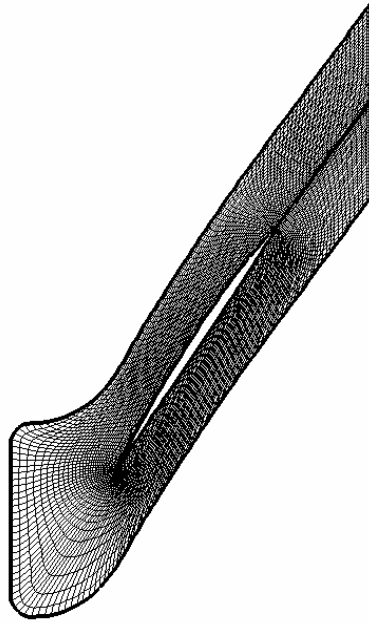
(c)  $L_\infty$  norm

**Figure 3.4** Variation of exact convergence error, error estimate, differences between two successive iterations and residuals for the Laplace problem on a  $80 \times 80$  Cartesian square grid with a relaxation parameter of 1.95.

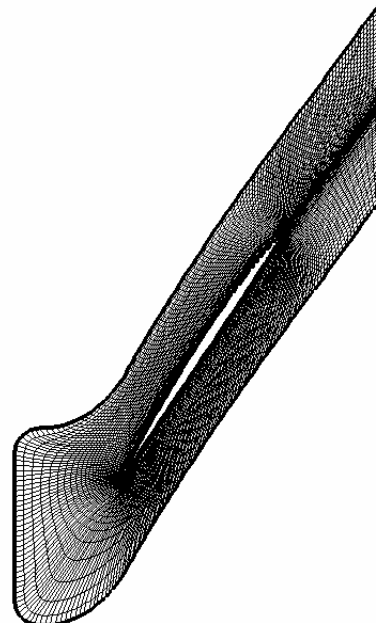
### 3.2.4 Application on a Euler and Navier-Stokes Solution of a Flow Problem

An Euler solver is applied on a turbomachinery cascade blade. Since, there is no exact solution for Euler and Navier-Stokes equations, exact solution is estimated at the best solutions achieved. To calculate exact solution code is run until the residual is reached to machine epsilon. At each iterations, difference between the computed solution and exact solution is regarded as the error.

Figure 3.5 shows the case compressor cascade geometries for Euler and Navier Stokes solutions. For both solution inlet and outlet Mach numbers are taken as 1.023 and 0.654 and corresponding Reynolds number is  $1.54 \times 10^6$ . For each solution, grid refinement is done for the first 100 iterations.



(a) Euler solution mesh 193x33



(b) Navier-Stokes solution mesh 257x49

**Figure 3.5** Solution Meshes

Both solutions are run for different number of CFL (Courant-Friedrichs-Lewy) numbers. The CFL number acts as the relaxation parameter. As the CFL number gets larger time steps gets larger and faster solution is achieved. For wave equation it is shown as;

$$c = \frac{u \cdot \Delta t}{\Delta x} \quad (3.57)$$

where  $\Delta t$  is the time step and  $u/\Delta x$  is the characteristic convection time. This number (also known as *Courant* number) represents the time required for a disturbance to be convected a distance  $\Delta x$ .

Euler solutions are represented in Figure 3.6 to Figure 3.8 for different norms and the CFL numbers. In these figures exact and estimated errors, changes between iterations, residuals and ratio parameter,  $r$ , given by equation (3.42) are shown. Ratio parameter is graphed only once for each case.

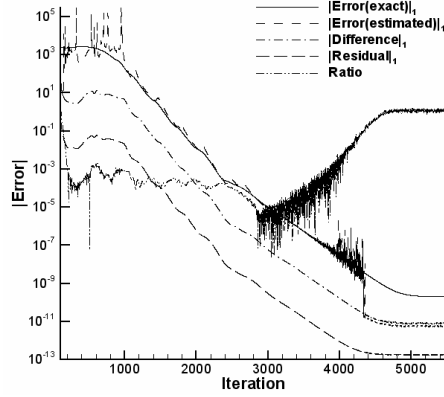
Figures 3.6 to 3.8 show Euler solution with CFL numbers of 3, 4 and 5, respectively. All three Euler solutions gave exactly the same results. Small values of ratio,  $r$ , showed that iterative scheme has real eigenvalues. For beginning 1000 iterations the iterative solution does not converge and estimation is oscillatory. The solution is not linearly decreasing until 3000 iterations and estimation is not that precise. When the solution goes linearly decreasing, ratio parameter,  $r$ , is gradually increasing. The estimation goes oscillatory when the changes between iterations begin to reach a limit of computer precision and ratio,  $r$ , reaches to the switch value of 0.1. This switch criteria of 0.1 did not work well that ratio reaches a value which is larger than the criteria and does not change; however, at these iterations solution has already converged. For the CFL number of 4, solution gave the slowest solution. For all cases  $L_1$  norm gave the best estimated results.

Figures 3.9 to 3.11 show Navier-Stokes solutions with the CFL numbers of 3,

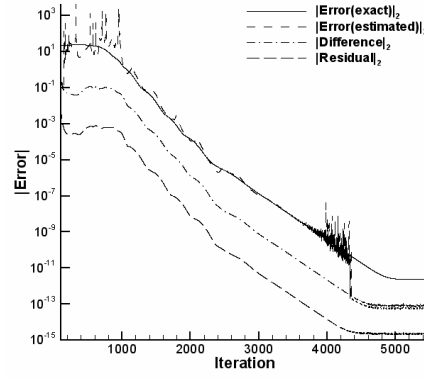
4 and 5, respectively. The solution is much more linearly decreasing with respect to Euler solution. From 800 iterations to the computer precision is reached, the estimation gave quite well results and solution decreased linearly. For the beginning iterations, convergence is not linearly decreasing and the estimation is oscillatory. Whenever linear decreasing is achieved in changes between successive iterations, the estimation gave good results. In the same manner in Euler solution, when the differences reached computer precision, estimation gave oscillatory results and ratio has reached a limit that is larger than the switch criteria. The ratio results show that again solution has real eigenvalues. Higher CFL numbers gave faster solutions. For higher CFL numbers the oscillations due to non-linear convergence have lasted for more number of iterations. In Figure 3.11 the oscillations reached almost 1500 iterations.

Convergence error estimation method has been used in Laplace, Euler and Navier-Stokes iterative solutions and gave quite well results. When linearly decreasing residuals are reached, convergence error estimations gave almost perfect results. However as the residuals approaches the smallest achievable quantities, estimations can give reliable numbers, so the ratio parameter,  $r$ , should be followed for reliability of this method.

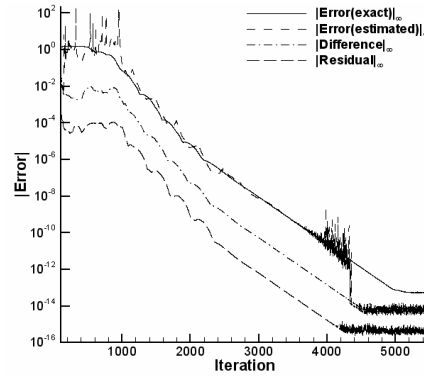




(a)  $L_1$  norm and ratio parameter,  $r$

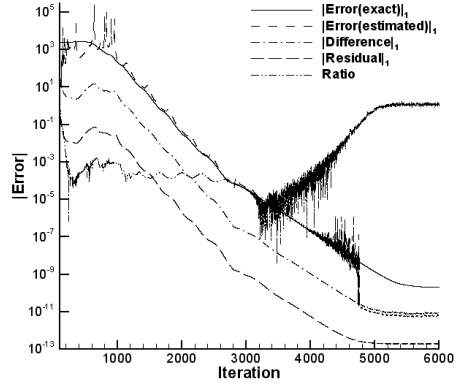


(b)  $L_2$  norm

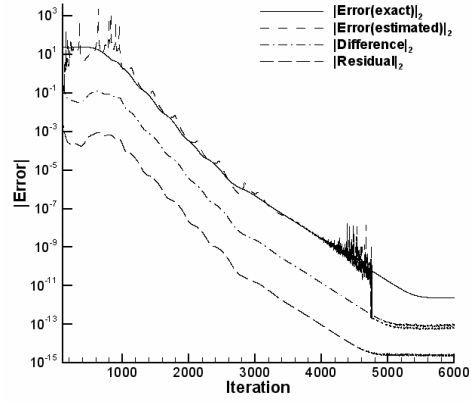


(c)  $L_\infty$  norm

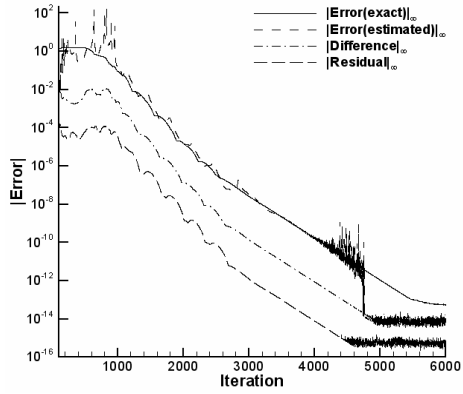
**Figure 3.6** Variation of norms of the exact convergence error, error estimate, differences, residual and ratio parameter for the Euler problem on a  $193 \times 33$  C-grid with CFL number of 3.



(a)  $L_1$  norm and ratio parameter,  $r$

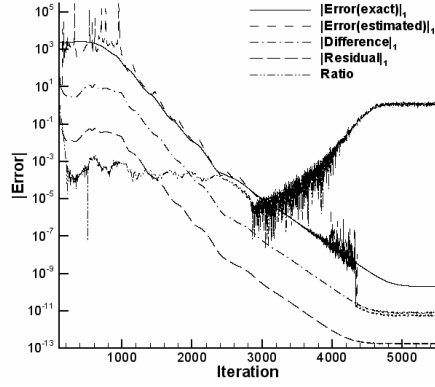


(b)  $L_2$  norm

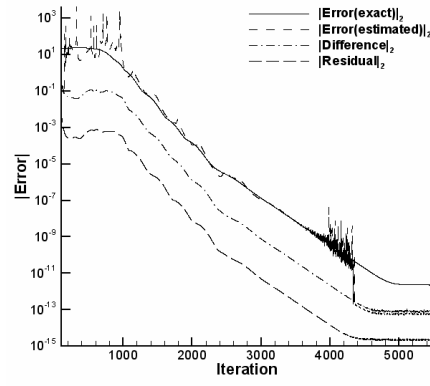


(c)  $L_\infty$  norm

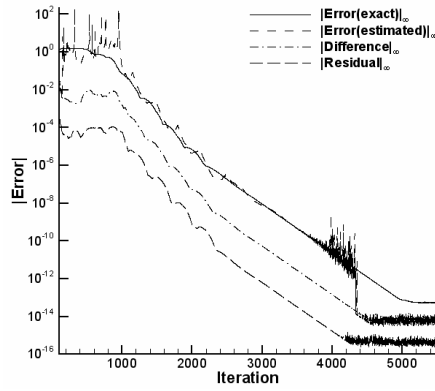
**Figure 3.7** Variation of norms of the exact convergence error, error estimate, differences, residual and ratio parameter for the Euler problem on a  $193 \times 33$  C-grid with CFL number of 4.



(a)  $L_1$  norm and ratio parameter,  $r$

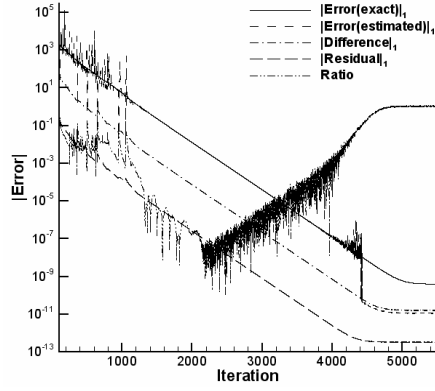


(b)  $L_2$  norm

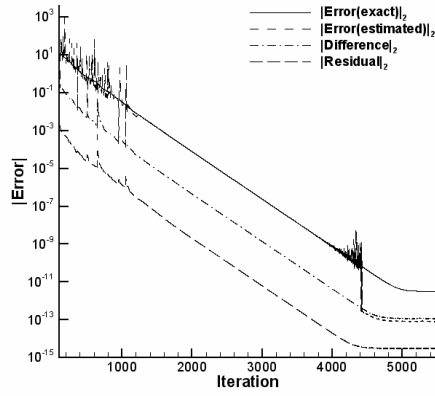


(c)  $L_\infty$  norm

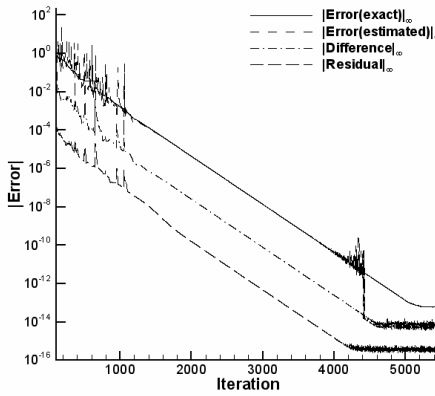
**Figure 3.8** Variation of norms of the exact convergence error, error estimate, differences, residual and ratio parameter for the Euler problem on a  $193 \times 33$  C-grid with CFL number of 5.



(a)  $L_1$  norm and ratio parameter,  $r$

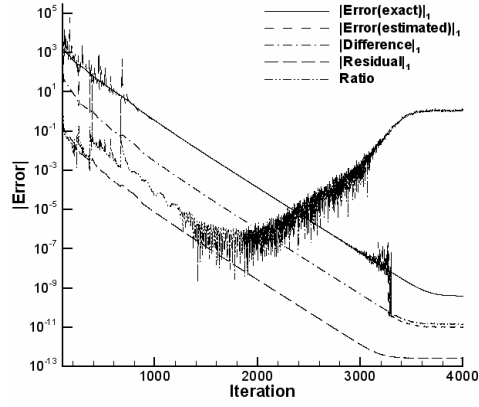


(b)  $L_2$  norm

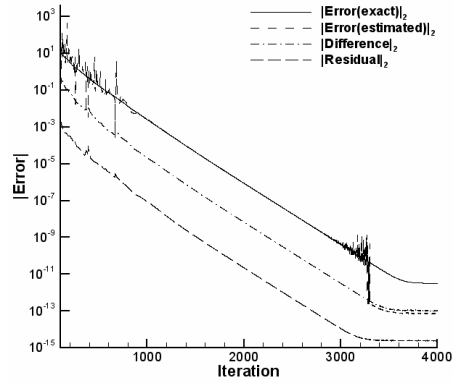


(c)  $L_\infty$  norm

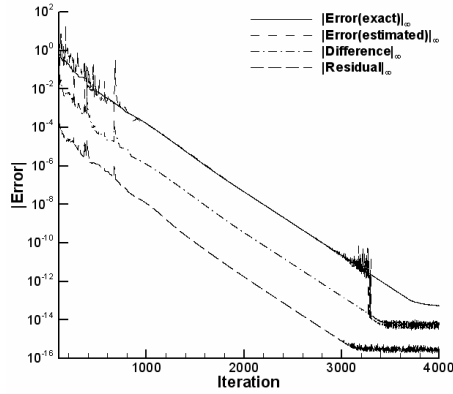
**Figure 3.9** Variation of norms of the exact convergence error, error estimate, differences, residual and ratio parameter for the Navier-Stokes problem on a  $257 \times 49$  C-grid with CFL number of 3.



(a)  $L_1$  norm and ratio parameter,  $r$

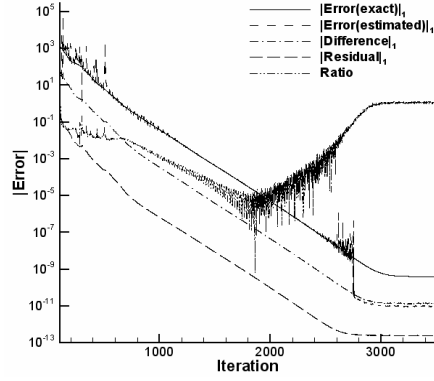


(b)  $L_2$  norm

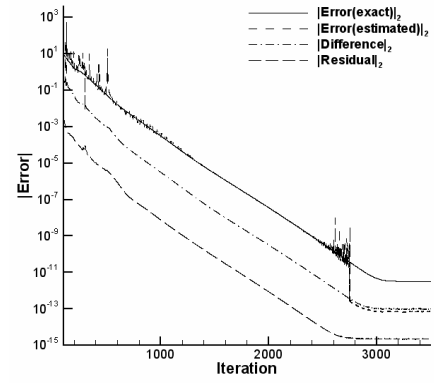


(c)  $L_\infty$  norm

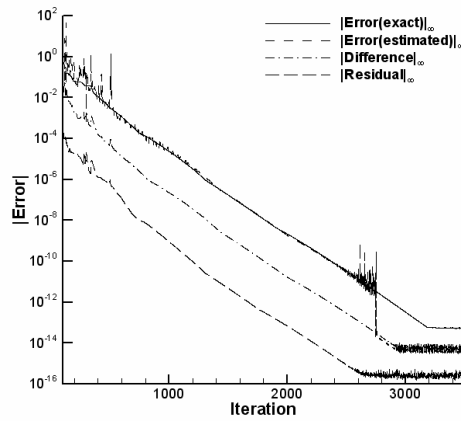
**Figure 3.10** Variation of norms of the exact convergence error, error estimate, differences, residual and ratio parameter for the Navier-Stokes problem on a  $257 \times 49$  C-grid with CFL number of 4.



(a)  $L_1$  norm and ratio parameter,  $r$



(b)  $L_2$  norm



(c)  $L_\infty$  norm

**Figure 3.11** Variation of norms of the exact convergence error, error estimate, differences, residual and ratio parameter for the Navier-Stokes problem on a  $257 \times 49$  C-grid with CFL number of 5.

### 3.3 Optimum Finite Difference Interval Estimation

In order to estimate the best finite difference interval that generates the minimum error in sensitivity calculations, a procedure is investigated. [2] This procedure is based on the approximation of the second derivative and the corresponding finite difference interval for accurate second derivative calculation. Acceptable values of second derivative are used for calculation of the optimum first order forward finite difference derivative formulation. Equation (2.48) is to be written as:

$$h_{forward} = 2\sqrt{\frac{|\mathcal{E}_{estimated}|}{|\Phi|}} \quad (3.58)$$

where  $h$  is the finite difference interval,  $|\mathcal{E}_{estimated}|$  is the norm of computation error and  $|\Phi|$  is the norm of second order derivative.

Procedure is based on the fact that the truncation error tends to be an increasing function of interval  $h$  while the cancellation error bound is generally a decreasing function of  $h$ . Throughout this procedure the value of  $h_{\Phi}$  is selected from a sequence of trials. The decision, whether a given value of  $\Phi$  is sufficiently accurate, is given by a defined cancellation error bound. The normalized value of equation (2.62) with second derivative value,  $|\Phi|$ , as;

$$|\hat{\mathcal{E}}_{cancellation\Phi}| = \frac{4|\mathcal{E}_{estimated}|}{h^2|\Phi|} \quad (3.59)$$

The interval for cancellation error norm,  $|\hat{\mathcal{E}}_{cancellation\Phi}|$ , is [0.001, 0.1]. For values below 0.001, truncation error tends to be large and for values over 0.1 cancellation error is large. It is assumed to be  $\Phi$  is not a large number and the value where  $\Phi$  is

calculate, is not a singularity. In this procedure, by trials  $|\Phi|$  and  $|\hat{\mathcal{E}}_{cancellation\Phi}|$  are calculated and checked. If  $|\hat{\mathcal{E}}_{cancellation\Phi}|$  is larger than the interval  $h_\Phi$  is increased, otherwise decreased.

The procedure not only checks the second derivative cancellation error, but also checks the forward and backward derivatives. Normalized cancellation errors for forward and backward derivatives are calculated by;

$$|\hat{\mathcal{E}}_{cancellation\ forward}| = \frac{2|\mathcal{E}_{estimated}|}{h \left| \frac{dw}{dX}_{forward} \right|} \quad (3.60)$$

$$|\hat{\mathcal{E}}_{cancellation\ backward}| = \frac{2|\mathcal{E}_{estimated}|}{h \left| \frac{dw}{dX}_{backward} \right|} \quad (3.61)$$

Procedure obtains optimum interval size  $h_\Phi$  by generating a sequence of trial values  $h_i$ . Also an initial guess for  $h_\Phi$  is required. Gill, Murray, Wright [15] suggest an initial guess for  $h_{forward}$  by a rough estimation of;

$$|\Phi| \approx 1 + |w| \quad (3.62)$$

then calculates initial guess of  $h_{forward}$  as;

$$\bar{h}_{forward} = 2 \sqrt{\frac{|\mathcal{E}_{estimated}|}{1 + |w|}} \quad (3.63)$$

and assumes that second order derivative optimum interval is one order larger than the first derivatives. Therefore the initial interval is:



$$h_0 = 10 \cdot \bar{h}_{forward} = 10 \cdot 2 \sqrt{\frac{|\mathcal{E}_{estimated}|}{1 + |w|}} \quad (3.64)$$

This procedure has been used for continuous functions. [15] This study adopts the procedure for domain problems and uses norms of the domains. The procedure can be stated as;

Step 1: (Initialization)

- Calculate  $|\mathcal{E}_{estimated}|$
- Calculate initial interval  $h_0 = 10 \cdot 2 \sqrt{\frac{|\mathcal{E}_{estimated}|}{1 + |w|}}$
- Evaluate  $w(X + h_0), w(X - h_0), \frac{dw}{dX}_{forward}(X + h_0), \frac{dw}{dX}_{backward}(X + h_0), \Phi(X, h_0), |\hat{\mathcal{E}}_{cancellation forward}|, |\hat{\mathcal{E}}_{cancellation backward}|, |\hat{\mathcal{E}}_{cancellation \Phi}|$ .
- Set  $k = 0$

Step 2: (Decide whether to accept, increase or decrease the interval)

- If  $\max\{|\hat{\mathcal{E}}_{cancellation forward}|, |\hat{\mathcal{E}}_{cancellation backward}|\} < 0.1$  and  $0.001 < |\hat{\mathcal{E}}_{cancellation \Phi}| < 0.1$ , set  $h_{\Phi} = h_0$ , go to step 5.
- If  $\max\{|\hat{\mathcal{E}}_{cancellation forward}|, |\hat{\mathcal{E}}_{cancellation backward}|\} < 0.1$  and  $\max\{|\hat{\mathcal{E}}_{cancellation forward}|, |\hat{\mathcal{E}}_{cancellation backward}|\} < 0.001$  go to step 4.
- Otherwise go to step 3.

Step 3: (Increase  $h$ )

Set  $k = k + 1$

Set  $h_k = 10 \cdot h_{k-1}$

- Evaluate  $w(X + h_0)$ ,  $w(X - h_0)$ ,  $\frac{dw}{dX}_{forward}(X + h_0)$ ,  $\frac{dw}{dX}_{backward}(X + h_0)$ ,  $\Phi(X, h_0)$ ,  $|\hat{\mathcal{E}}_{cancellation\ forward}|$ ,  $|\hat{\mathcal{E}}_{cancellation\ backward}|$ ,  $|\hat{\mathcal{E}}_{cancellation\ \Phi}|$ .
- If  $\max\{|\hat{\mathcal{E}}_{cancellation\ forward}|, |\hat{\mathcal{E}}_{cancellation\ backward}|\} < 0.1$  and  $0.001 < |\hat{\mathcal{E}}_{cancellation\ \Phi}| < 0.1$  set  $h_\Phi = h_0$ , go to step 5.
- If  $k = k_{\max}$  stop.
- Otherwise repeat step 3.

Step 4: (Decrease  $h$ )

- Set  $k = k + 1$
- Set  $h_k = h_{k-1} / 10$
- Evaluate  $w(X + h_0)$ ,  $w(X - h_0)$ ,  $\frac{dw}{dX}_{forward}(X + h_0)$ ,  $\frac{dw}{dX}_{backward}(X + h_0)$ ,  $\Phi(X, h_0)$ ,  $|\hat{\mathcal{E}}_{cancellation\ forward}|$ ,  $|\hat{\mathcal{E}}_{cancellation\ backward}|$ ,  $|\hat{\mathcal{E}}_{cancellation\ \Phi}|$ .
- If  $\max\{|\hat{\mathcal{E}}_{cancellation\ forward}|, |\hat{\mathcal{E}}_{cancellation\ backward}|\} > 0.1$  or  $|\hat{\mathcal{E}}_{cancellation\ \Phi}| > 0.1$ , set  $h_\Phi = h_{k-1}$ , go to step 5.
- If  $0.001 < |\hat{\mathcal{E}}_{cancellation\ \Phi}| < 0.1$ , set  $h_\Phi = h_k$ , go to step 5.
- If  $k = k_{\max}$  stop.
- Otherwise repeat step 4.

Step 4: (Compute the estimate of the optimal interval)

- Define  $h_{forward}$  by the equation;

$$h_{forward} = 2\sqrt{\frac{|\mathcal{E}_{estimated}|}{|\Phi|}}$$

- Set  $\mathcal{E}_{Total\ forward}$  by the equation;

$$\mathcal{E}_{Total\ f'_{forward}} = \frac{h_{forward}}{2}|\Phi| + \frac{2 \cdot |\mathcal{E}_{estimated}|}{h_{forward}}$$

### 3.3.1 Case Study I: Laplace Solution

A case study is performed for a Laplace solution for  $w(x, y, t) = t^4 100xy$  as described in section 3.2. Two different grids of 40x40 and 80x80 are used. Iterations are stopped with different criteria that define the stopping residual values of  $10^{-6}$ ,  $10^{-8}$ ,  $10^{-10}$  and  $10^{-12}$ . Three different norms of  $L_1$ ,  $L_2$  and  $L_\infty$  are used. Above described optimum finite difference step size procedure is applied and solution is repeated for different intervals and errors are graphed for verification. On the graphs, estimated norm values are shown.

In figure 3.12, norms of exact error, truncation error and estimated cancellation error variations are represented for 40x40 grid. Estimated optimum finite difference interval value and corresponding estimated total error is pointed out on the graph. All the estimated values are at the junction of truncation error and the estimated cancellation error curves. The most accurate results are estimated with the  $L_1$  norms. Therefore, following results are graphed by  $L_1$  norms.

Figure 3.13 and Figure 3.14 shows the results in  $L_1$  norm for 40x40 and 80x80 grids, respectively. Optimum interval size estimations are more precise with the

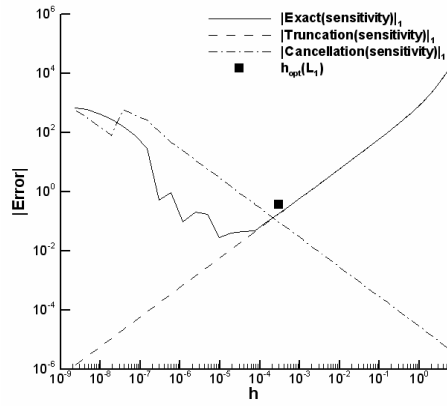
estimated cancellation error on fine grid of  $80 \times 80$ . In general procedure gave approximate results in one order of magnitude due to the approximations of cancellation error estimations. Numerical results are tabulated in table 3.1 for  $40 \times 40$  grid solution and in table 3.2 for  $80 \times 80$  grid solution.

**Table 3.1** Sensitivity error, step size and error estimations for Laplace solution  $w(x, y, t) = t^4 100xy$  on  $40 \times 40$  grid

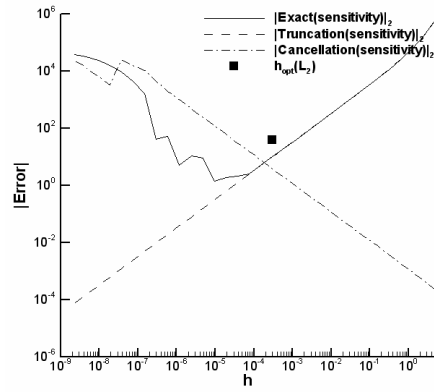
Convergence	$ h_{opt} _1$	$ e_{estimated} _1$	$ e_{actual} _1$	$ h_{opt} _2$	$ e_{estimated} _2$	$ e_{actual} _2$	$ h_{opt} _\infty$	$ e_{estimated} _\infty$
$10^{-6}$	$3,03 \cdot 10^{-4}$	0,365	0,182	$2,98 \cdot 10^{-4}$	38,9	19,8	$2,54 \cdot 10^{-4}$	1,22
$10^{-8}$	$3,05 \cdot 10^{-5}$	$3,66 \cdot 10^{-2}$	$1,83 \cdot 10^{-2}$	$3,00 \cdot 10^{-5}$	3,91	1,99	$2,55 \cdot 10^{-5}$	0,123
$10^{-10}$	$3,01 \cdot 10^{-6}$	$3,61 \cdot 10^{-3}$	$1,81 \cdot 10^{-3}$	$2,96 \cdot 10^{-6}$	0,385	0,196	$2,52 \cdot 10^{-6}$	$1,21 \cdot 10^{-2}$
$10^{-12}$	$3,30 \cdot 10^{-7}$	$3,96 \cdot 10^{-4}$	$1,99 \cdot 10^{-4}$	$3,17 \cdot 10^{-7}$	$4,14 \cdot 10^{-2}$	$2,16 \cdot 10^{-2}$	$2,87 \cdot 10^{-7}$	$1,38 \cdot 10^{-3}$

**Table 3.2** Sensitivity error step size and error estimations for Laplace solution  $w(x, y, t) = t^4 100xy$  on  $80 \times 80$  grid

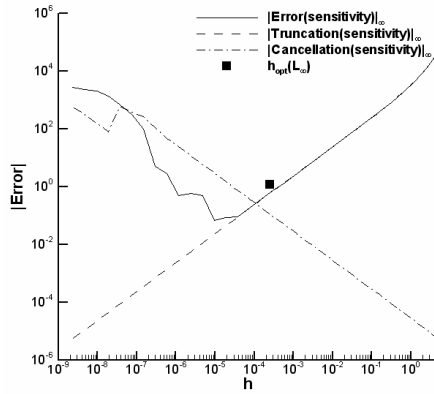
Convergence	$ h_{opt} _1$	$ e_{estimated} _1$	$ e_{actual} _1$	$ h_{opt} _2$	$ e_{estimated} _2$	$ e_{actual} _2$	$ h_{opt} _\infty$	$ e_{estimated} _\infty$
$10^{-6}$	$2,19 \cdot 10^{-4}$	0,263	0,131	$2,15 \cdot 10^{-4}$	14,3	7,26	$1,84 \cdot 10^{-4}$	0,884
$10^{-8}$	$2,17 \cdot 10^{-5}$	$2,61 \cdot 10^{-2}$	$1,30 \cdot 10^{-2}$	$2,13 \cdot 10^{-5}$	1,42	0,721	$1,83 \cdot 10^{-5}$	$8,77 \cdot 10^{-2}$
$10^{-10}$	$2,21 \cdot 10^{-6}$	$2,65 \cdot 10^{-3}$	$1,33 \cdot 10^{-3}$	$2,17 \cdot 10^{-6}$	0,144	$7,34 \cdot 10^{-2}$	$1,86 \cdot 10^{-6}$	$8,92 \cdot 10^{-3}$
$10^{-12}$	$2,24 \cdot 10^{-7}$	$2,69 \cdot 10^{-4}$	$1,34 \cdot 10^{-4}$	$2,18 \cdot 10^{-7}$	$1,45 \cdot 10^{-2}$	$7,44 \cdot 10^{-3}$	$1,92 \cdot 10^{-7}$	$9,23 \cdot 10^{-4}$



(a)  $L_1$  norm

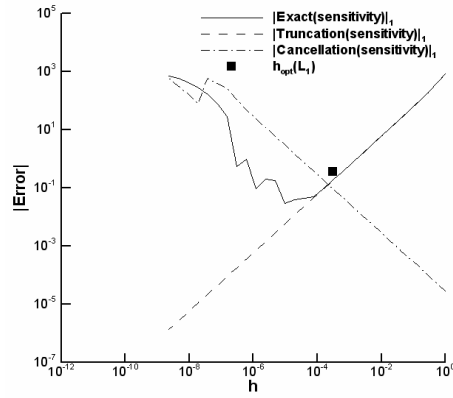


(b)  $L_2$  norm

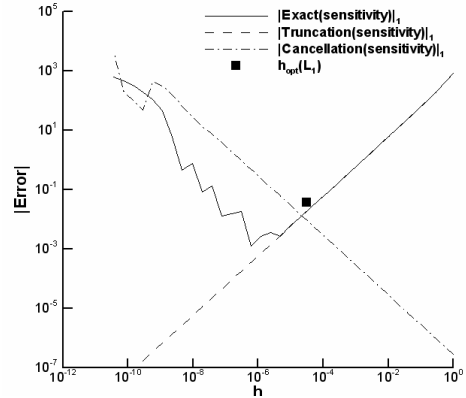


(c)  $L_\infty$  norm

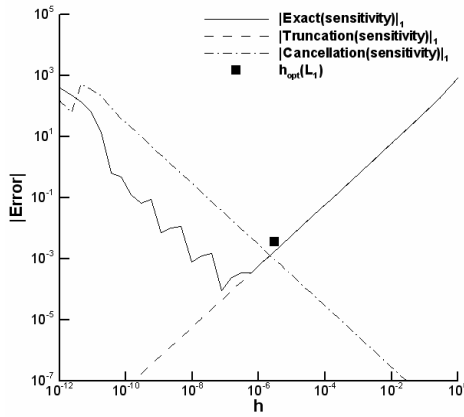
**Figure 3.12** Error variations of sensitivities and corresponding estimated optimum intervals on 40x40 grid Laplace solution for different norms.



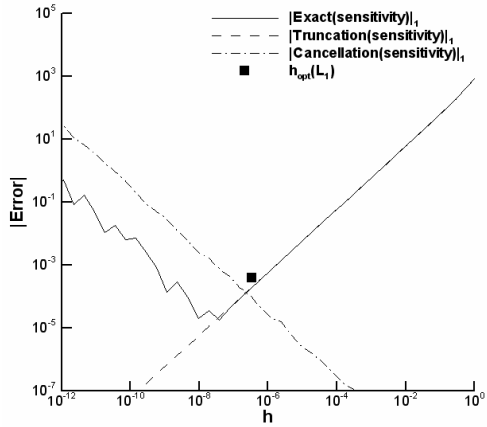
(a)  $10^{-6}$



(b)  $10^{-8}$

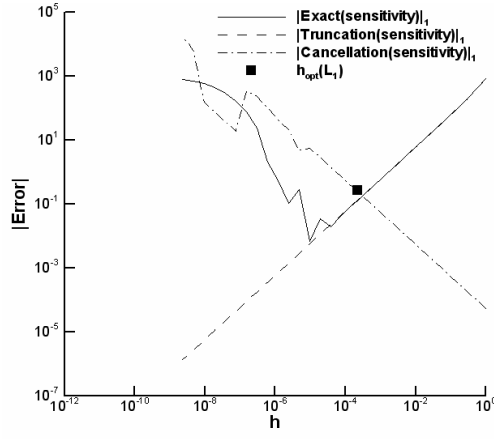


(c)  $10^{-10}$

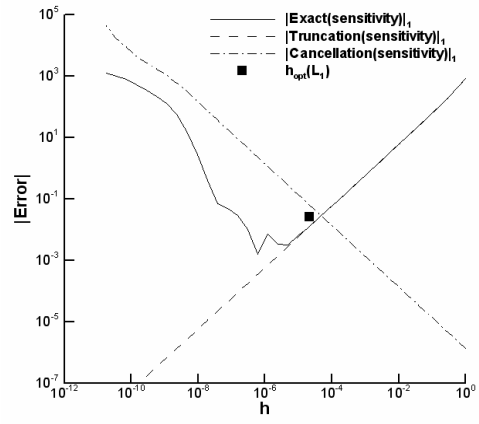


(d)  $10^{-12}$

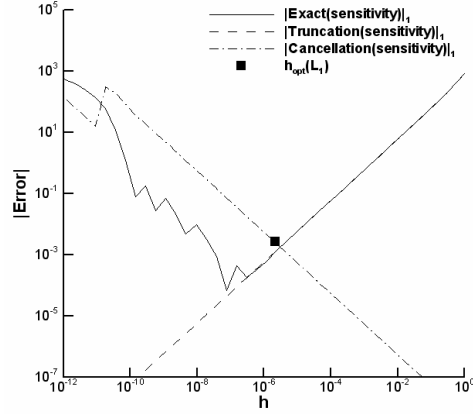
**Figure 3.13** Error variations of sensitivities and corresponding estimated step sizes on 40x40 grid Laplace solution for  $L_1$  norms of different iteration stopping criteria.



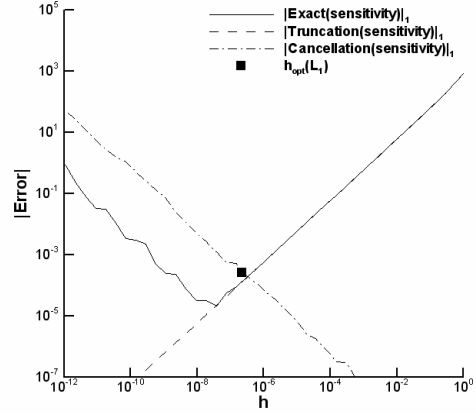
(a)  $10^{-6}$



(b)  $10^{-8}$



(c)  $10^{-10}$



(d)  $10^{-12}$

**Figure 3.14.** Error variation of sensitivities and corresponding estimated step sizes on 80x80 grid Laplace solution for  $L_1$  norms of different iteration stopping criteria.



### 3.3.2 Case Study II: Euler Solution

Optimum finite difference interval procedure is applied to the Euler inverse design algorithm. The exact errors that are calculated by the differences between numerical iterations and analytical solution are calculated. Interval size that generates the minimum exact error is referred as the optimum interval. The above described procedure is applied and the results are compared. Also a comparison for full inverse design cycle including optimization is included. Inverse design algorithm is explained in Chapter 4.

The design practices are performed for the Rotor R 030. [26] A solution mesh is given in figure 3.1.(a). The flow condition is set to outlet Mach number of 1.1, inlet Mach number of 1.4, flow angle of  $56.8^\circ$  and Reynolds number of  $10^6$ . Algorithm, briefly gets target and base geometries, defines perturbations on base geometry and calculates the sensitivities of the flow variables. Sensitivities are inputted to an optimization algorithm that tries to reach the target values and a new geometry is designed. Pressure distribution on the design geometry is compared with the target's by a convergence parameter:

$$CP = \left( \frac{\sum_{i=1}^I (P_{t_i} - P_{b_i})^2 \Delta S_i}{\sum_{i=1}^I \Delta S_i} \right)^{\frac{1}{2}} \quad (65)$$

where  $P_t$  and  $P_b$  are the target and baseline pressures respectively on the cascade surface at point  $i$  and  $\Delta S_i$  is the length of the surface element.

Figures 3.15 to 3.18 show the exact error variation with Patched and Wagner polynomials. (Appendix) In figures 3.15 to 3.17, a patched polynomial is used for perturbation where the perturbation centers are 10%, 50% and 90% of the chord length, respectively. In all cases, three different iteration termination criteria are

used. Iterations are terminated with respect to estimated errors reached  $10^{-4}$ . In figures errors are graphed in three different norms of  $L_1$ ,  $L_2$  and  $L_3$ .

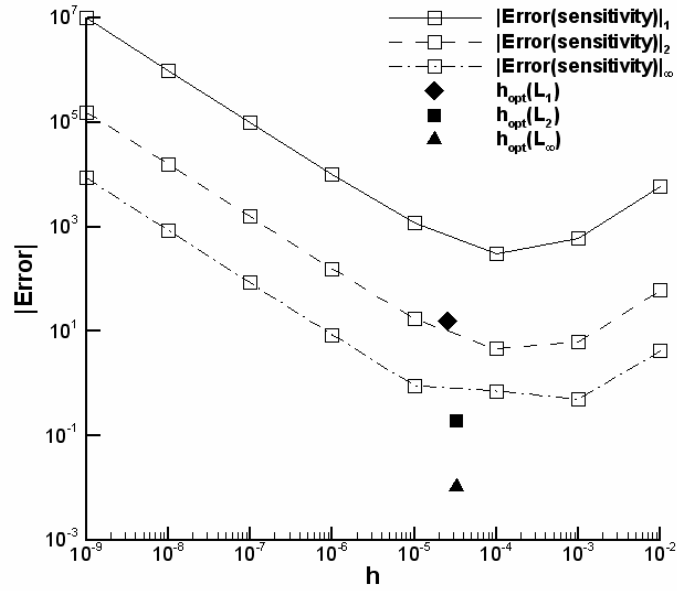
The calculated optimum interval estimations are pointed out in the graphs. The estimations are in one order of magnitude precise. In three cases estimations are smaller than the optimum value and in figure 3.16 it is larger than the optimum. Error amount differences between the calculated interval's and the real optimum are in one order of magnitude. There's no notable difference between the usage of different norms.

In figure 3.16 it is hard to distinguish the optimum value of interval, because error quantities are almost the same between  $10^{-3}$  to  $10^{-7}$ . Exact error quantities are calculated as the differences of flow variables that found with the finite difference sensitivity algorithm and the ones calculated with the analytical sensitivity algorithm. The algorithms used in this study did not give the same results. Total error amounts are all less than the exact value. The terms in equation (2.47) are approximate values. Total error is calculated estimated values of interval,  $h$ , second derivative  $|\Phi|$  and error,  $\varepsilon$ .

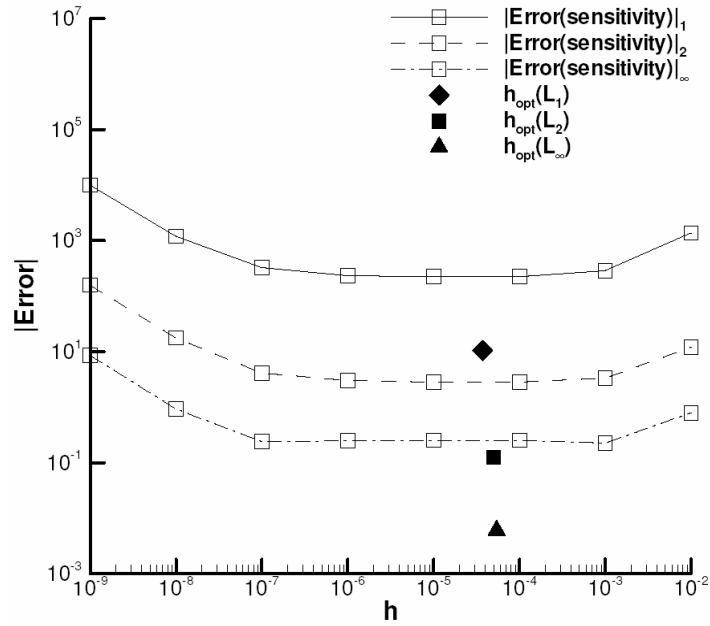
This method is used in sample full design cycle. In this case, 10 patched polynomials are defined as 5 each on upper and lower surfaces at 0.15, 0.30, 0.50, 0.70 and 0.85 stations. For each perturbation an optimum perturbation size (finite difference interval) is calculated by the above described method. Design cycle is repeated 10 times but the perturbation sizes are only defined in the first cycle.

Figure 3.19 and 3.20 shows the pressure and density contours of the flow domains, respectively. Also, geometries and surface pressure distributions are compared in figure 3.21 and 3.22. The results are almost identical. This design is repeated with different given perturbation amounts. In these solutions perturbation amounts are pre-defined in the input. Figure 3.23 shows convergence parameter variation with respect to perturbation sizes. Small perturbations of  $10^{-8}$ ,  $10^{-9}$  and  $10^{-10}$

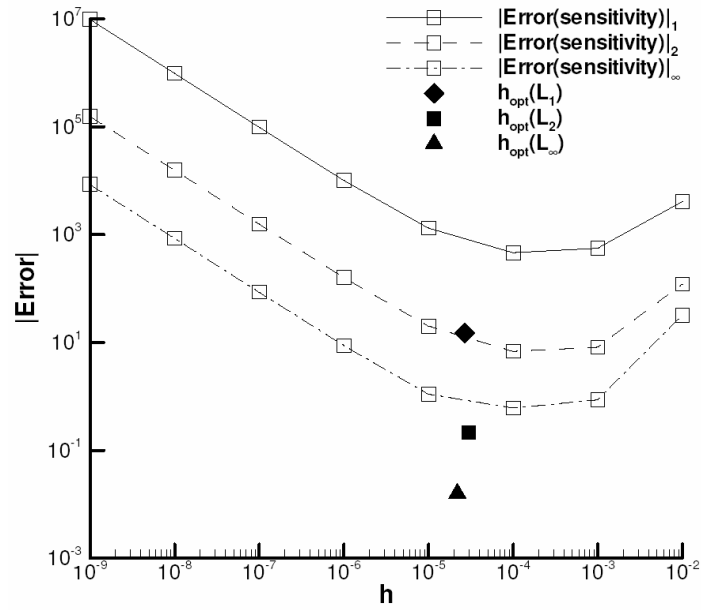
gave quite larger quantities for convergences and also larger perturbations of  $10^{-2}$  and  $5 \times 10^{-1}$  gave larger quantities for convergences. Optimum perturbations calculated by the method are averaged and corresponding convergence value is pointed out in figure 3.23. Convergence parameters over the design cycles are also graphed in figure 3.24.



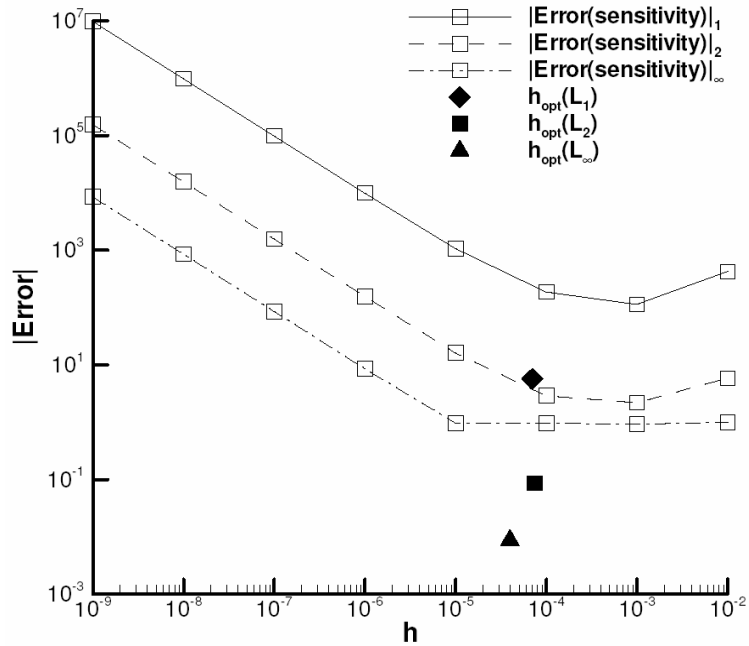
**Figure 3.15** Error variations of sensitivities vs. finite difference interval size  $h$  with Patched polynomial bump of 0.1/1.0 station on upper surface. Estimated optimum interval sizes and corresponding estimated errors are indicated also represented.



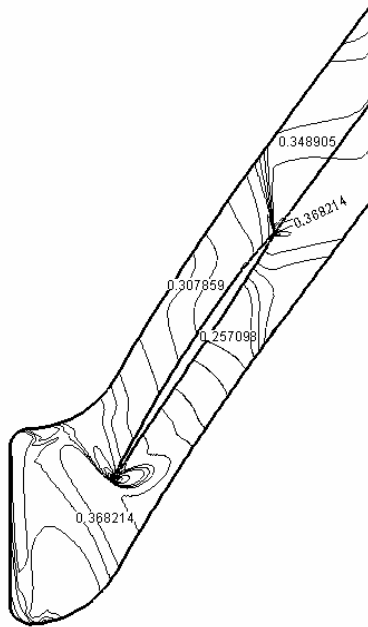
**Figure 3.16** Error variations of sensitivities vs. finite difference interval size  $h$  with Patched polynomial bump of 0.5/1.0 station on upper surface. Estimated optimum interval sizes and corresponding estimated errors are indicated also represented.



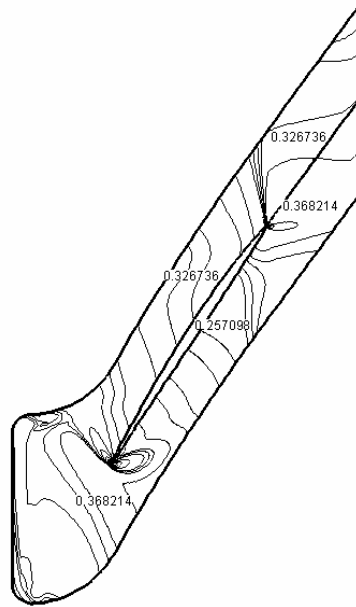
**Figure 3.17** Error variations of sensitivities vs. finite difference interval size  $h$  with Patched polynomial bump of 0.9/1.0 station on upper surface. Estimated optimum interval sizes and corresponding estimated errors are indicated also represented.



**Figure 3.18** Error variations of sensitivities vs. finite difference interval size  $h$  with Wagner Polynomials Perturbations on upper surface. Estimated optimum interval sizes and corresponding estimated errors are indicated also represented.

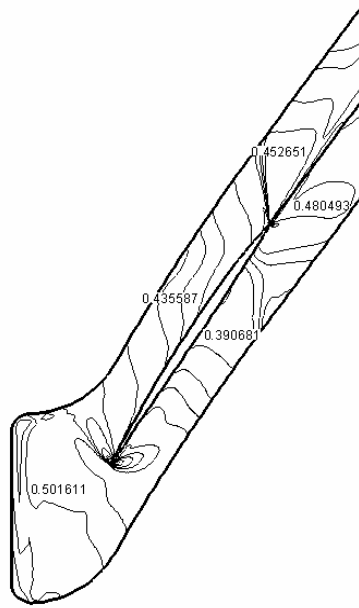


(a) Pressure Contour - Design

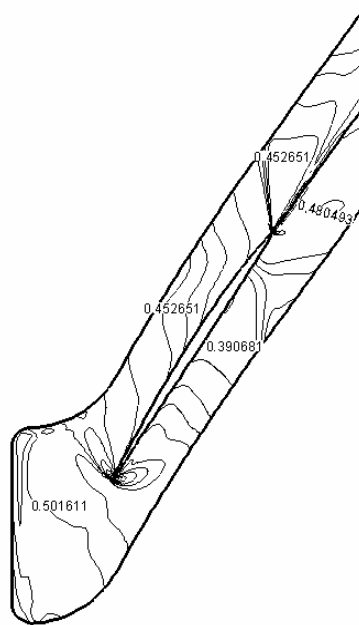


(b) Pressure Contour - Target

**Figure 3.19** Pressure distributions of design and target geometries under the same flow conditions.

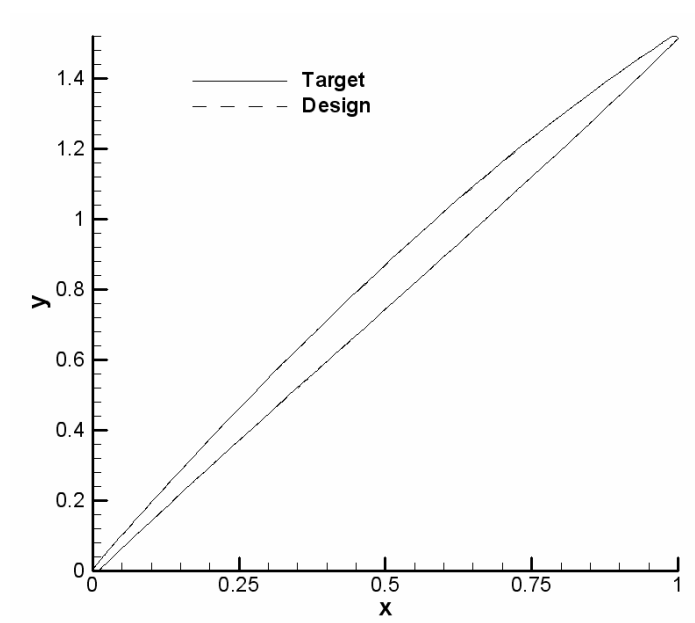


(a) Density Contour - Design

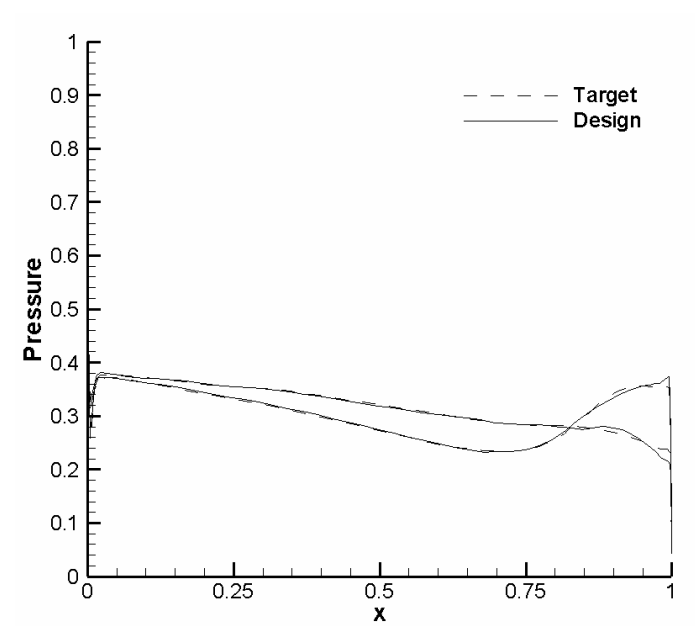


(b) Density Contour - Target

**Figure 3.20** Density distributions of design and target geometries under the same flow conditions.

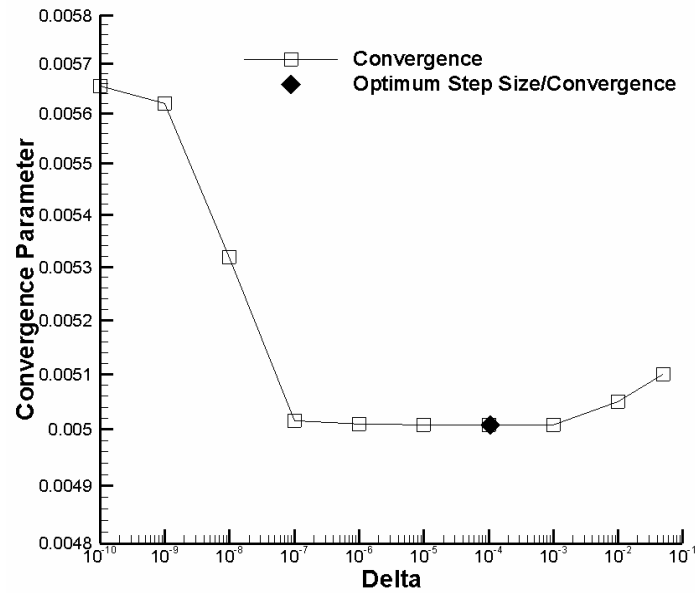


**Figure 3.21** Target and Design geometries

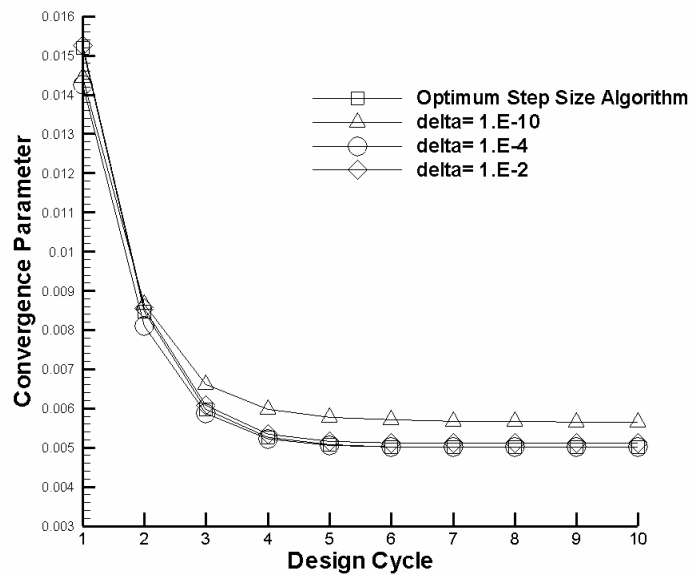


**Figure 3.22** Target and Design surface pressure distributions





**Figure 3.23** Convergence Parameter variations vs. surface perturbation delta. Design is completed with 10 design cycles with Patched polynomials.



**Figure 3.24** Convergence Parameter variations vs. design cycle. Design is completed with 10 design cycles with Patched polynomials.

## **CHAPTER 4**

### **EFFICIENCY IMPROVEMENTS**

#### **IN FINITE DIFFERENCE SENSITIVITY ANALYSIS**

##### **4.1 Introduction**

In inverse design algorithm, a base geometry is perturbed and sensitivities are calculated for each perturbation. In this study, accuracy and efficiency of the finite-difference sensitivity derivatives are investigated. Most of the computation time is used for finite difference sensitivity calculations. These calculations are independent tasks and derivative values are inputted into an optimization process. Increasing efficiency of the design procedure is investigated by parallel computing of the finite difference sensitivity derivatives. Theory and application of parallel computing are investigated below and results for an example application are discussed.

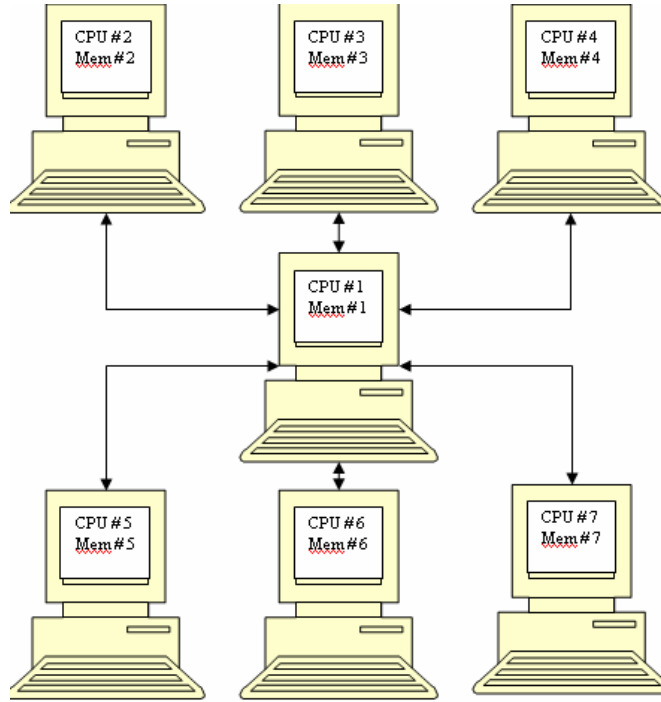
##### **4.2 Parallel Computing**

Parallel processing, the method of having many small tasks solve one large problem, has emerged as a key enabling technology in modern computing. The past several years have witnessed an increasing acceptance and adoption of parallel processing for high-performance scientific computing. This enabled lower cost and sustained productivity. The acceptance has been facilitated by two major developments: massively parallel processors (MPPs) and the widespread use of distributed computing.

MPPs are powerful computers that combine a few hundred to a few thousand CPUs in a single large cabinet connected to large memories. MPPs are generally used for global climate modeling and drug design. As simulations become more realistic, the computational power required to produce them grows rapidly. Thus, researchers on the cutting edge turn to MPPs and parallel processing in order to get the most computational power possible.

The second major development affecting scientific problem solving is distributed computing. Distributed computing is a process whereby a set of computers connected by a network are used collectively to solve a single large problem. As more and more organizations have high-speed local area networks interconnecting many general-purpose workstations, the combined computational resources may exceed the power of a single high-performance computer. The most important factor in distributed computing is cost. Large MPPs typically cost more than \$10 million but distributed computing systems offers very little cost in contrast especially on a local set of existing computers.

In all parallel processing, data must be exchanged between cooperating tasks. The message-passing model is generally used from the perspective of the number and variety of multiprocessors that support it, as well as in terms of applications, languages, and software systems that use it. The Parallel Virtual Machine (PVM) system uses the message-passing model to allow programmers to distribute computing across a wide variety of computer types, including MPPs. A key concept in PVM is that it makes a collection of computers appear as one large virtual machine. Figure 4.1 represents a cluster of computers that runs the different tasks on different CPUs over the main computer using PVM software.



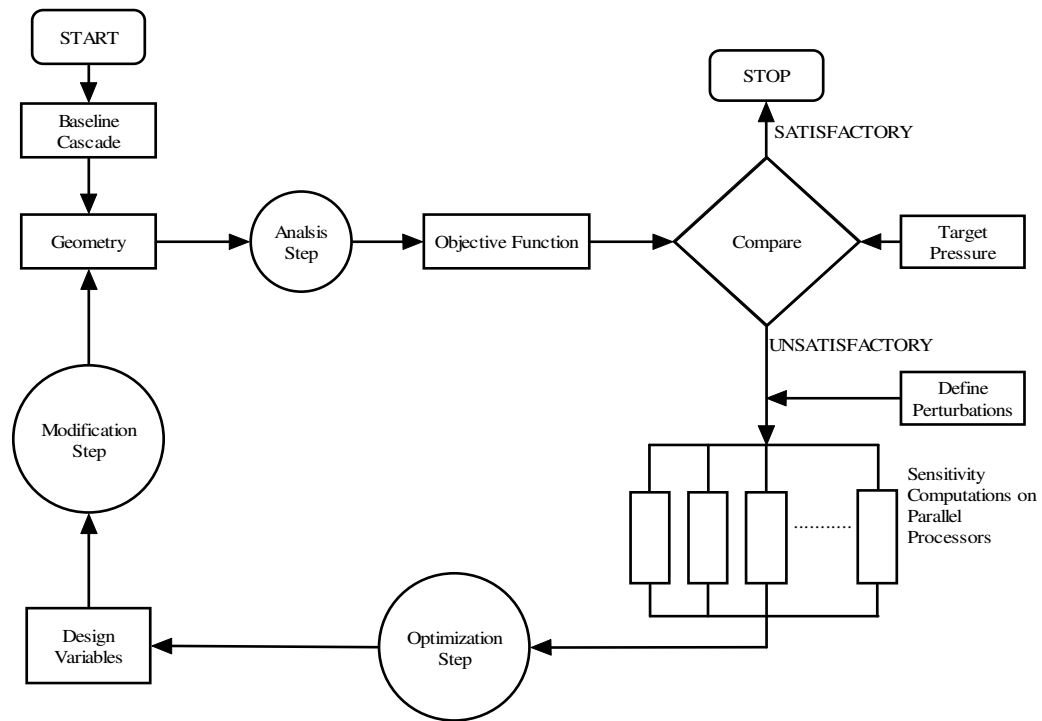
**Figure 4.1** Cluster of CPUs connected from separate computers by PVM software in order to form a single virtual computer with multiple processors.

### 4.3 Inverse Design Algorithm

In an inverse design algorithm, a base geometry is perturbed and sensitivities are calculated for each perturbation. In this study sensitivities are calculated by finite difference methods. Each finite difference sensitivity calculation is run independently for every perturbation. In several studies, number of independent perturbations varies between 8 and 16. [16] In order to decrease computational time inverse design algorithm can be arranged that each finite difference sensitivity calculations can be defined as independent tasks. Arranged algorithm is shown in Figure 4.2.

This algorithm runs only the finite difference sensitivity calculations on parallel computers. Analyses of optimization and design geometries are calculated on the main processor, flow analyses of each perturbation and the corresponding finite

difference sensitivity calculations are done on different processors. Main processor waits until all sensitivity values are calculated in order to go optimization step.



**Figure 4.2** Parallel Inverse Design Algorithm

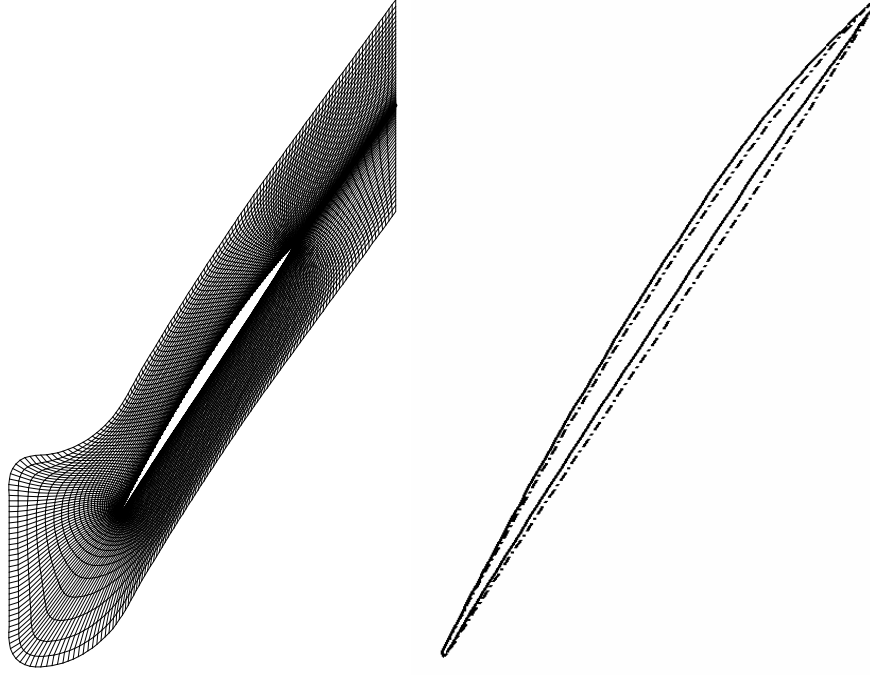
#### 4.4 Case Study

Comparison of process times between a single and multiple processors is performed. The complete design cycle is done with different perturbation sizes and shape functions. Code input values and boundary conditions stayed same. The time is measured in minutes by the time measuring standard subroutines. 16 perturbations are defined at the beginning and sensitivity calculations are calculated on 1, 2, 4, 8 and 16 CPUs (PIII 750 MHz). Results are grouped for 3 different shape functions

and 5 different perturbation sizes. The shape functions are Wagner, Hicks-Henne and patched polynomials (Appendix) The perturbation sizes are 0.0005, 0.00075, 0.001, 0.00125 and 0.0015.

The design code performed Navier-Stokes equations for flow analyses. The flow analysis is based on the solution of 2-D, unsteady, compressible Navier-Stokes equations over a curvilinear co-ordinate system. The Baldwin Lomax eddy viscosity model is used for turbulence closure, and the transition point is fixed at fourteen percent of the chord. A finite volume method is employed for the spatial discretization. The flow variables are defined at cell centers, and centered differencing is used for the spatial derivatives. Second and fourth-order artificial viscosities are added to enforce numerical stability. The time integration is performed using an explicit, four-stage Runge-Kutta scheme. Local time stepping, variable-coefficient implicit residual smoothing, and a multigrid method are implemented to accelerate the convergence. Characteristic boundary conditions are imposed at the far-field boundary based on a one-dimensional eigenvalue analysis, and a no-slip, adiabatic-wall conditions are used on the cascade surface.

The design practices are performed for the Rotor R 030. [26] Computational structured grids of  $257 \times 49$  are generated. The minimum grid spacing next to the cascade surface is set to .001 percent of the chord length. The profiles at 150 and 190 mm. in radial axis are used as target and baseline cascades, respectively. Figure 4.3 shows the computational grids for target and base shapes. The profile at 150 mm. in radial axis is located at mid-span of the blade. The flow condition is set to  $M_2 = 0.654$ ,  $M_1 = 1.023$ ,  $\alpha = 56.8^\circ$  and  $Re = 1.54 \times 10^6$ . Here,  $M_2$ ,  $M_1$ ,  $\alpha$  and  $Re$  are the outlet and inlet Mach numbers, inlet flow angle and Reynolds number, respectively.



**Figure 4.3** 257x49 Computational Grid for Rotor R 030 blade [26] and Geometrical Presentation of Target (dashed) and Base (solid) airfoils

In order to judge the design quality and monitor the convergence of the design cycle, a convergence parameter, CP, is defined. This parameter is based on the root-mean-square of length-weighted pressure discrepancies between the target pressure and the pressure of the designed cascade:

$$CP = \left( \frac{\sum_{i=1}^I (P_{t_i} - P_{b_i})^2 \Delta S_i}{\sum_{i=1}^I \Delta S_i} \right)^{\frac{1}{2}} \quad (4.1)$$

where  $P_t$  and  $P_b$  are the target and baseline pressures respectively on the cascade surface at point  $i$  and  $\Delta S_i$  is the length of the surface element. There are total of  $I$  elements on the cascade.

## 4.5 Results

Results are tabulated in Table 4.1. The execution time decreased with increasing number of processors. Gain in execution time showed an asymptotic decreasing trend with the number of processors. Because, only flow analyses for finite-difference sensitivity calculations are parallel run, all other computations are performed on a single processor. The communication time between the processors is insignificant compared to total design execution time. Ratio of processing time of multiple processors to the single is called as speed-up and tabulated in Table 4.2. Tabulated data, designed shape and surface pressures are graphed in Figure 4.4, 4.5 and 4.6 for Wagner, Hicks-Henne and Patched polynomials, respectively.



Table 4.1 Execution Time (min) and Convergences

(a) Wagner Functions

#CPU	delta=0.0015	delta=0.00125	delta=0.001	delta=0.00075	delta=0.0005
1	471	347	380	339	323
2	262	190	212	184	178
4	183	121	137	116	112
8	170	79	94	76	74
16	103	57	70	53	56
Convergence	0.00542	0.00849	0.00224	0.00219	0.00213

b) Hicks-Henne Functions

#CPU	delta=0.0015	delta=0.00125	delta=0.001	delta=0.00075	delta=0.0005
1	464	476	581	406	446
2	353	267	312	237	256
4	262	180	205	191	165
8	169	127	144	137	106
16	120	99	102	105	78
Convergence	0.00824	0.00849	0.00835	0.00679	0.00723

(c) Patched Polynomials

#CPU	delta=0.0015	delta=0.00125	delta=0.001	delta=0.00075	delta=0.0005
1	317	352	403	386	304
2	174	191	188	180	174
4	110	123	114	109	102
8	75	81	72	70	66
16	58	57	51	51	48
Convergence	0.00671	0.00671	0.00366	0.00670	0.00680

Table 4.2 Parallel Speed-up

(a) Wagner Functions

#CPU	delta=0.0015	delta=0.00125	delta=0.001	delta=0.00075	delta=0.0005
1	1.00	1.00	1.00	1.00	1.00
2	1.80	1.83	1.79	1.84	1.81
4	2.57	2.87	2.77	2.92	2.88
8	2.77	4.39	4.04	4.46	4.36
16	4.57	6.09	5.43	6.40	5.77

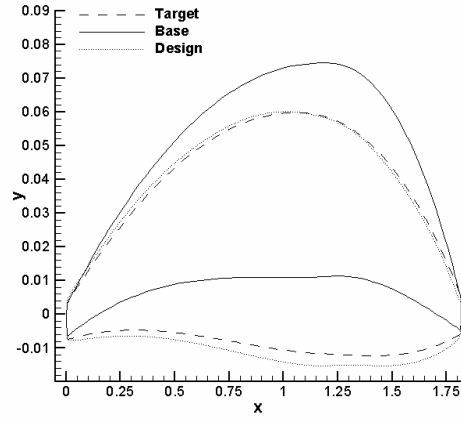
(b) Hicks-Henne

Functions

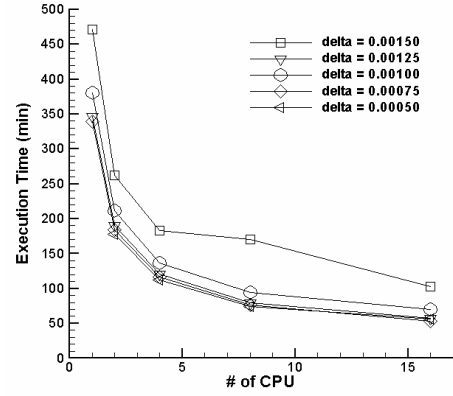
#CPU	delta=0.0015	delta=0.00125	delta=0.001	delta=0.00075	delta=0.0005
1	1.00	1.00	1.00	1.00	1.00
2	1.31	1.78	1.86	1.71	1.74
4	1.77	2.64	2.83	2.13	2.70
8	2.75	3.75	4.03	2.96	4.21
16	3.87	4.81	5.70	3.87	5.72

(c) Patched Polynomials

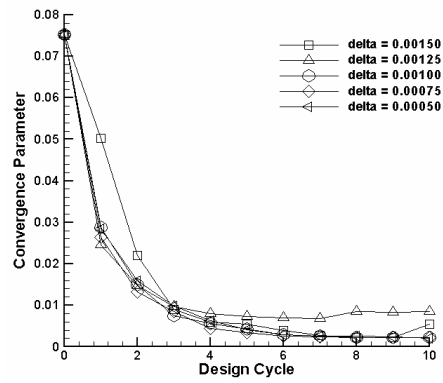
#CPU	delta=0.0015	delta=0.00125	delta=0.001	delta=0.00075	delta=0.0005
1	1.00	1.00	1.00	1.00	1.00
2	1.82	1.84	2.14	2.14	1.75
4	2.88	2.86	3.54	3.54	2.98
8	4.23	4.35	5.60	5.51	4.61
16	5.47	6.18	7.90	7.57	6.33



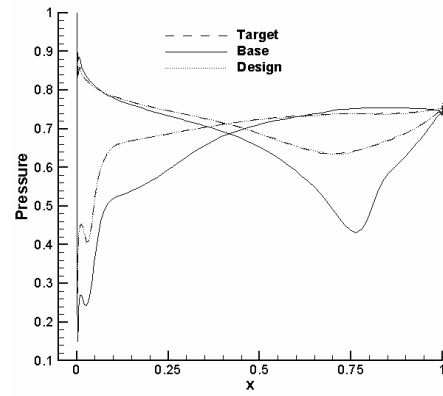
(a) Evolution of Blade Geometry



(b) Evolution of Surface Pressure

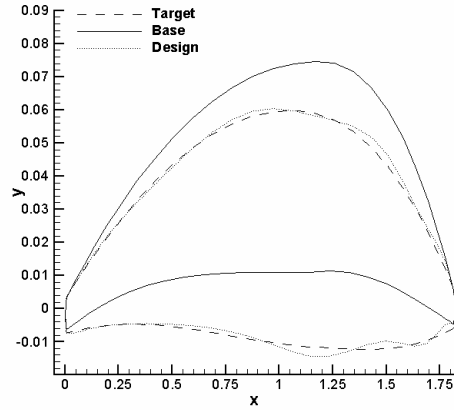


(c) Convergence History

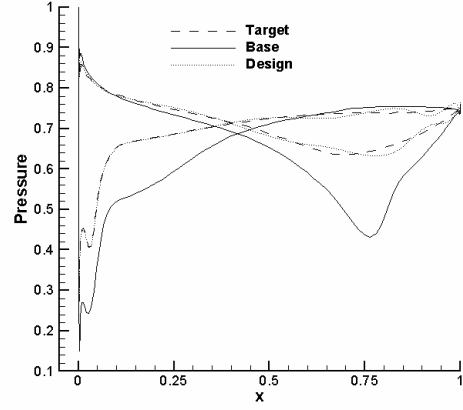


(d) CPU times

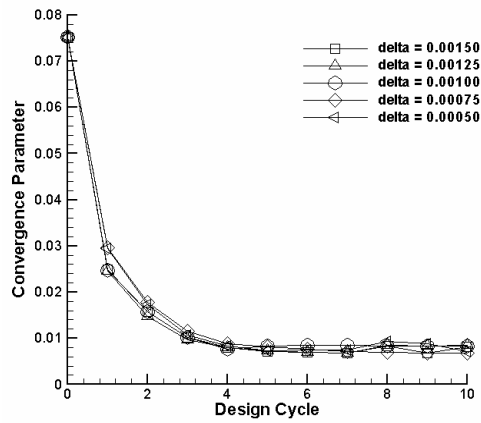
Figure 4.4 Design Practice with 16 Wagner Functions



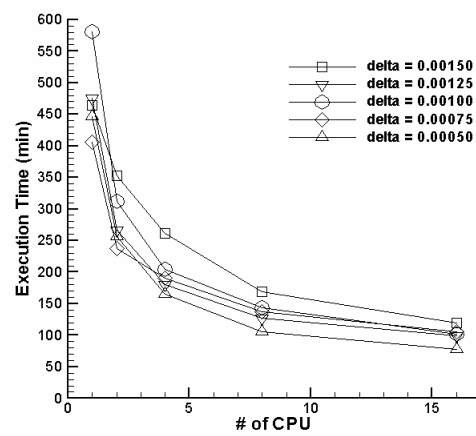
(a) Evolution of Blade Geometry



(b) Evolution of Surface Pressure

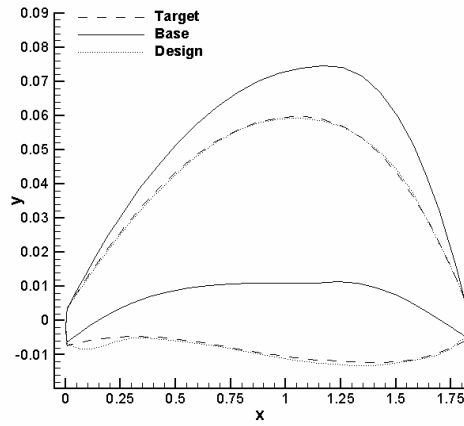


(c) Convergence History

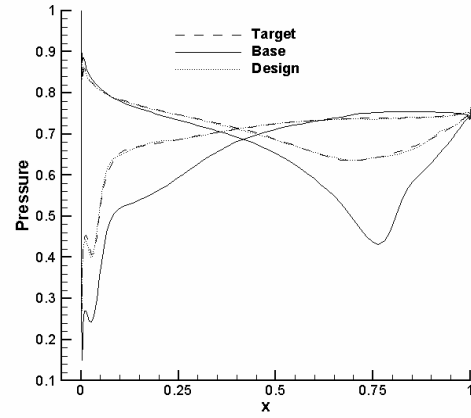


(d) CPU times

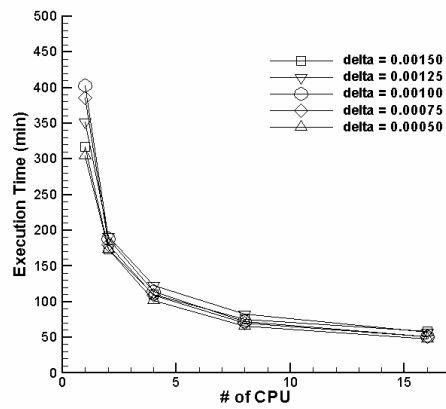
Figure 4.5 Design Practice with 16 Hicks-Henne Functions



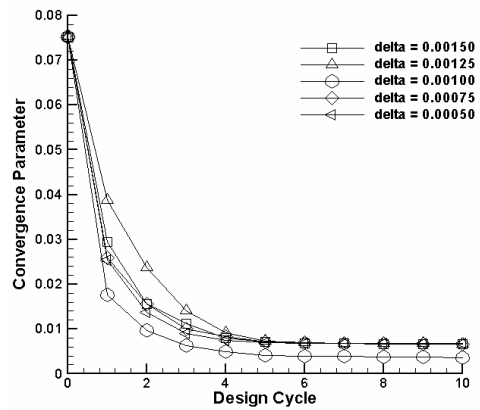
(a) Evolution of Blade Geometry



(b) Evolution of Surface Pressure



(c) Convergence History



(d) CPU times

Figure 4.6. Design Practice with 16 Patched Polynomials

## 4.6 Discussions

In this inverse design algorithm, most of the processing time is used for finite difference sensitivity calculations. Each sensitivity analysis performs a flow solution with the new perturbed geometry. Flow solution begins from the last updated solution; however, convergence is still not reached after a quite large number of iterations. Some perturbations may sometimes cause more changes in flow environment than expected such as changing shapes occurs on shock generated areas, leading edge or trailing edge. Therefore some sensitivity calculations can last longer than others and whole process waits for it.

Gain in execution time showed an asymptotic decreasing trend with the number of processors. Because, only flow analyses for finite-difference sensitivity calculations are run parallel, all other computations are performed on a single processor. Also different processing times due to perturbation differences slow down the process. The communication time between the processors is insignificant compared to total design execution time. Ideally number of processors should be equal to number of independent tasks, but, regarding to the cost of hardware reasonable decrease can be preferred with less number of processors.

With this case study, effect of shape functions and perturbation sizes can also be compared. Wagner functions gave the most converged solutions in overall but in advancing side of the blade shape made an off-set due to limited convergence at sharp edges. Hicks-Henne functions caused wavy surface designs. Patched polynomials gave reasonable solutions for most of the geometry but an abnormal bump stayed on the designed geometry. In validation of the designs, convergence parameter is calculated by pressure differences, but shapes also be checked out for abnormalities. For sharp edges, it is hard to design; so, shape functions should be capable of creating edges.

Convergences differ with changing perturbation sizes. This was expected, since it is known that there exists an optimum step size that minimizes the total error in finite difference sensitivity calculations. Truncation error decreases with decreasing step size of a finite difference derivative, but cancellation error increases with decreasing step size. This study also described the importance of the importance of accuracy improvements in sensitivity calculations.

## **CHAPTER 5**

### **CONCLUSION AND RECOMMENDATIONS**

Sensitivity calculation is one of the most important parts in the design optimization. The performance of finite difference sensitivities was studied. Compared to analytical method, finite difference methods are easy to use in sensitivity calculation. However finite difference sensitivities have accuracy problems and computational time may be very large.

Accuracy improvements in finite difference sensitivities are investigated by error analysis. One of the ways to reduce the error in finite difference sensitivity calculation is to use optimum finite difference interval size. Finding optimum interval size requires an estimation of convergence error and estimation of second order sensitivities. Convergence errors are analyzed and it has been represented that the construction of the iteration scheme directly affects the convergence error. The relation between the eigenvalues of iteration matrices and error propagation are investigated and a method is applied. The presented convergence error estimation method, which is based on eigenvalue analysis, is successful in predicting convergence error.

In this method convergence errors are predicted for both real and complex eigenvalues. Results show that estimating convergence error with real eigenvalues was successful. A parameter is stated for decision whether the eigenvalues of iteration matrix is complex or not. The switching parameter that distinguishes real and complex eigenvalue may not be very reliable in some problems. A better method



can be developed for estimating the type of eigenvalues or switching criteria should be identified by general rules.

The optimum finite difference interval size method requires value of second derivative. Second derivative estimation is represented by a method that makes trials due to error estimations. Usage of estimated values and trials make this method rather inefficient. Therefore; estimating second order derivative is difficult; but may be very useful in calculating optimum interval size. Approximating second derivative as 1.0 may not be useful because order of magnitude of second derivative may be very large like in this study.

Efficiency of the finite difference sensitivities can be improved by parallel computing. Modifying code for parallel sensitivity calculation does not require large modification in usual code. This study used an algorithm that a processor behaves as central and uses parallel processors in sensitivity calculations. Initial and design analyses, optimization processes are still done on a central processor but independent tasks of finite difference sensitivity calculations, which takes most of the computation time, are spread over parallel processors. Therefore; number of processors does not linearly decrease the computation time and half the number of tasks are recommended for number of processors.

For future studies, it can be said that the accuracies in finite difference step sizes highly depend on the second derivative estimations. In this study second derivative estimation has been done over a trial method. This way increases the computational time and did not give excellent results. Convergence error estimation method is highly recommended for iterative solutions; however, the control on eigenvalues of iteration matrix should be done followed that eigenvalues can go complex values for some cases. Aid of parallel computing should be taken when finite difference sensitivity analysis is performed in an inverse aerodynamic design algorithm. The computational time can be decreased in great amounts. Inverting a code to a parallel algorithm does not take some much effort and modifications.

## REFERENCES

- [1] Anderson J.D., A History of Aerodynamics, Cambridge, 1997
- [2] Iott, J., Haftka, R. T. and Adelman, H. M. "Selecting Step Sizes in Sensitivity Analysis by Finite Differences", *NASA Technical Memorandum 86382*, 1985.
- [3] Gill, P. E., Murray, W., and Wright M. H., *Practical Optimization*, Academic Press, New York, 1981.
- [4] Barton, R.R., "Computing Forward Difference Derivatives in Engineering Optimization", *Eng.Opt.*, Vol.20, pp.205-224, 1992
- [5] Kirsch, U., Bogomolni, M., "Efficient Finite Difference Design Sensitivities", *AIAA Journal*, Vol.43, No.2, February 2005
- [6] Sung, C. and Kwon, J.H., "Accurate Aerodynamic Sensitivity Analysis Using Adjoint Equations", *AIAA Journal*, Vol.38, No.2, February 2000
- [7] Kocabiçak, E. and Eyi, S., "Effects of Sensitivity Analysis on Turbomachinery Blade Design", *AIAA 200-3361*, 36<sup>th</sup> AIAA/ASME/SAE/ASEE Joint Propulsion Conference, 2000
- [8] Ferziger, J.H. and Peric, M., *Computational Methods for Fluid Dynamics*, 3<sup>rd</sup> edition: Springer, 2002

- [9] Ferziger, J.H. and Peric, M., “Further discussions of Numerical Errors in CFD”, *International Journal for Numerical Methods in Fluids*, Vol. 23, 1263-1274, 1996
- [10] Hoffman, J., *Numerical Methods for Engineers and Scientists*, McGraw-Hill, 1992
- [11] Ferziger, J.H., *Numerical Methods for Engineering Applications*, Wiley, Chichester, 1998
- [12] Kaplan, B., “Performance Evaluation and Improvements of Inverse Compressor Cascade Design”, M.S. Thesis, METU-AEE, 2000
- [13] Kocabıçak, E., “Sensitivity Analysis in Turbomachinery Cascade Design Optimization”, M.S. Thesis, METU-AEE, 2000
- [14] *Computational Methods for Aerodynamic Design and Optimization*, AGARD Conference Proceedings No. 463, May, 1989
- [15] *Special Course on Inverse Methods for Airfoil Design for Aeronautical and Turbomachinery Applications*, AGARD Report No.780, May 1990
- [16] Sobieczky, H., “Progress in Inverse Design and Optimization in Aerodynamics,” *Computational Methods for Aerodynamic Design and Optimization*, AGARD Conference Proceedings No. 463, May, 1989.
- [17] Dulikravich, G.S., “Aerodynamic Shape design and Optimization: Status and Trends”, *Journal of Aircraft*, Vol.29, No. 6, Nov-Dec.1992
- [18] Çetinkaya, T. A., Akmandor I. S. and Ucer A. S., “A Computational Design Method for Shock Free Transonic Cascades and Airfoils,” *Computational Methods for Aerodynamic Design and Optimization*, AGARD Conference Proceedings No. 463, May, 1989.

- [19] Vicini A and Quagliarella D., “Inverse and Direct Airfoil Design Using a Multiobjective Genetic Algorithm,” AIAA Journal, Vol. 35, No. 9, September 1997.
- [20] Geist, A. Beguelin, A., Dongarra, J., Jiang, W., Manchek, R. and Sunderam, V., PVM 3 User’s Guide and Reference Manual, Oak Ridge National Laboratory, September 1994.
- [21] Special Course on Parallel Computing in CFD, AGARD Report No.807, October 1995.
- [22] Anderson, J.D., *Fundamentals of Aerodynamics*, McGraw-Hill, Intl.Ed. 1991
- [23] Hoffmann, A. K. and Chiang, S. T., *Computational Fluid Dynamics for Engineers Volume I,II* , Publication of Engineering System, Austin, Texas 78713.
- [24] Baldwin, B. and Lomax, H., “Thin Layer Approximation and Algebraic Model for Separated Flows,” AIAA Paper 78-257, January 1978.
- [25] Haftka, R.T., Gürdal, Z., *Elements of Structural Optimization*, Springer, 3rd edition, 2002
- [26] Dunker R. Test Case E/CO-4: Single Transonic Compressor Stage. *AGARD Advisory Report No:275*, pp. 245-285, 1990.

## APPENDIX

### DESIGN VARIABLES

#### Cascade Shape Modification

The cascade geometry was modified by adding smooth perturbations. The geometry perturbation  $\Delta y$  was defined as a linear combination of base functions  $f_k$ .

$$\Delta y(x) = \sum_{k=1}^K X_k f_k(x) \quad (\text{A.1})$$

where  $X_k$  is the perturbation magnitude, or design variable,  $f_k$  is the shape function, and  $K$  is the number of design variables to be used.

#### Wagner Functions:

Wagner functions are frequently used in optimization. They provide large variations with high harmonics and may cause waviness in resulting designs.

$$f_1(x) = \frac{[\theta + \sin(\theta)]}{\pi} - \sin^2\left(\frac{\theta}{2}\right) \quad (\text{A.2})$$

$$f_k(x) = \frac{\sin(k\theta)}{k\pi} + \frac{\sin[(k-1)\theta]}{\pi} \quad \text{for } k > 1$$

where

$$\theta = 2 \sin^{-1}(\sqrt{x})$$

**Patched Polynomials:**

A cubic on one side of  $x_k$  is patched with another cubic on the other side to produce a smooth curve of second-order continuity.  $x_k$  is the location of maximum perturbation.

$$f_k(x) = 1 - \left( \frac{x - x_k}{x_k} \right)^2 \left( 1 + \frac{A}{(1 - x_k)^2} \frac{x}{x_k} \right) \quad \text{for } 0 \leq x \leq x_k \quad (\text{A.3})$$

$$f_k(x) = 1 - \left( \frac{x - x_k}{1 - x_k} \right)^2 \left( 1 + \frac{B}{(x_k)^2} \frac{1 - x}{1 - x_k} \right) \quad \text{for } x_k \leq x \leq 1$$

where

$$A = \max(0, 1 - 2x_k)$$

$$B = \max(0, 2x_k - 1)$$

**Hicks-Henne Functions:**

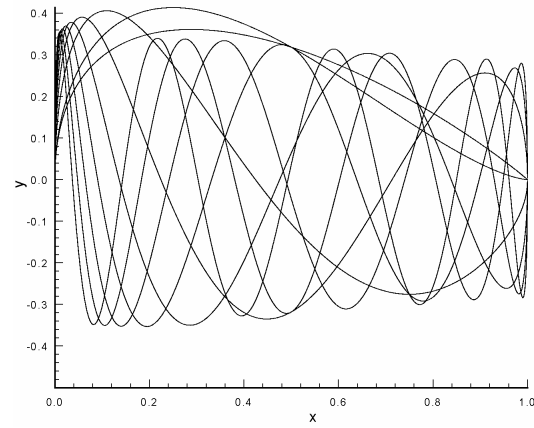
The sinusoidal shape functions are frequently used in airfoil optimization.

$$f_k(x) = \sin^3(\pi x^{e(k)}) \quad (\text{A.4})$$

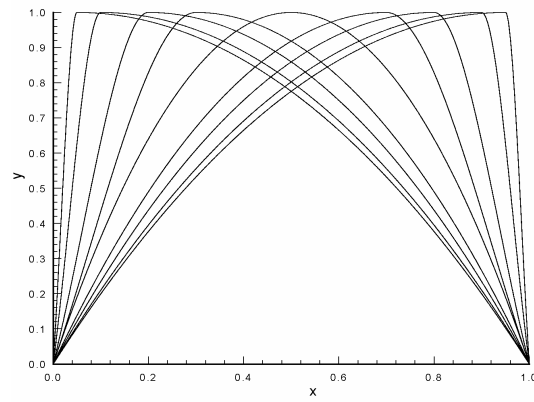
where

$$e(k) = \frac{\log(0.5)}{\log(x_k)}$$

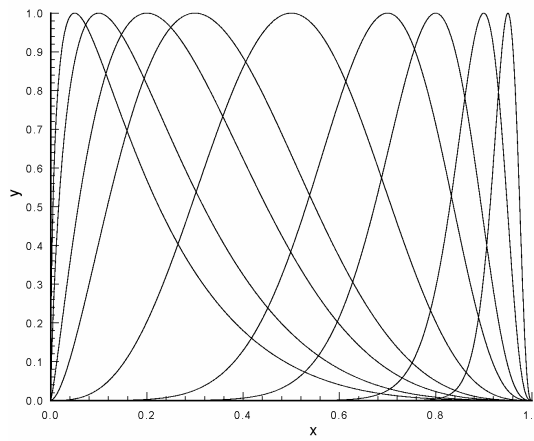
Here  $x_k$ 's are the locations of maximum height of the corresponding shape functions.



(a) Wagner functions



(b) Patched polynomials



(c) Hicks-Henne functions

**Figure A.1** Shape functions used to perturb the geometry