Stability Analysis and Stabilization of Unstructured Finite Volume Method

by

Liang Chen

BEng, Northwestern Polytechnical University (Xi'an, China), 2012

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

The Faculty of Graduate and Postdoctoral Studies

(Mechanical Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

April 2016

Abstract

In this thesis, we develop a stability analysis model for the unstructured finite volume method. This model employs the matrix method to implement stability analysis. For the full discretization, where the temporal discretization employs backward Euler time-stepping, a linearization is used to construct the model. Both for the full discretization and semi-discretization, the stability condition is expressed in terms of eigenvalues. The validity of the stability analysis model is verified for linear cases and nonlinear cases. The analysis in this thesis also explains the phenomena that the defined energy can locally increase in the energy stability analysis method. This model can be applied to both linear problems and nonlinear problems; in this thesis, we focus on the 2D Euler equations.

We also develop a stabilization methodology. This methodology is aimed at optimizing the eigenvalues of the Jacobian matrix by changing the coordinates of interior vertices of a mesh. Specifically, for an unstable spatial discretization, we can shift the unstable eigenvalues into stability region by changing the mesh. The stabilization methodology is verified by numerical cases as well. The success of this stabilization relies on a developed method to change the eigenvalues of a matrix in a quantitative and controllable way. This method is a general approach to optimize the eigenvalues of a matrix, which means it can be applied to other systems as well.

Preface

The topic of the research in this thesis was chosen by Dr. Carl Olliver-Gooch. The research for this thesis was implemented by the author Liang Chen under the supervision of Dr. Carl Olliver-Gooch. The manuscript preparation was done by Liang Chen with the guidance from Dr. Carl Olliver-Gooch.

Table of Contents

Appendices

List of Tables

List of Figures

List of Symbols

Roman Symbols

$\bar{U}$ The solution vector of the discretized system or the computational solution vector

$\bar{U}_p$ The computational solution after perturbation

$\bar{U}_s$ The solution of the discretized system and the fixed point

Δt The time step

Δx The space step in the x-direction

δx The adjustment for the x-coordinate of one vertex in the mesh

δy The adjustment for the y-coordinate of one vertex in the mesh

$\Delta \bar{U}$ The computational error

$\delta \bar{U}$ The solution update

$\Delta \bar{U}_p$ The perturbation to the solution

$\frac{\partial \bar{R}}{\partial \bar{U}}$ The Jacobian matrix

$\hat{n}$ The outward unit normal vector

d Dimension

E Numerical energy

E Total energy

$E_{\Delta\lambda_0}$ The relative error of the predicted eigenvalue change

F The flux vector

G The amplification factor defined in the von Neumann method

I The identity matrix and the imaginary unit

L_u The unit length at a vertex

R Real number

$R(\bar{U})$ The residual term

S The source terms in Poissson's equation

S The surface of the control volume

T The period of the amplification factor associated with the case where the largest eigenvalues are a conjugate pair

U The conservative variables vector

u Velocity in the x-direction

V The control volume

v Velocity in the y-direction

x The right eigenvector of a matrix

y The left eigenvector of a matrix

P The pressure

Greek Symbols

$\bar{\gamma}$ The average amplification factor

$\Delta\gamma$ The difference between the actual amplification factor and the norm of the largest eigenvalue

$\Delta\theta$ The difference between the actual angle and π or 0

$\Delta\zeta$ The vector of coordinate change

ε The perturbation of a coordinate of one vertex in the mesh

γ The amplification factor

Γ_k The ratio of the norm of the computational error at the kth iteration to that of the 0th iteration

λ Eigenvalue of a matrix

Ω The spatial domain

$\partial\Omega$ The boundary of the spatial domain

ρ Density

θ The angle associated with the real part and the imaginary part of an eigenvalue

ζ The vector of the coordinates of vertices of a mesh

Superscripts

n time level

Subscripts

i Index

j Index

Acknowledgements

Here I first would like to thank Dr. Carl Olliver-Gooch for his constant patient supervision and constant tolerance on different views. His insightful strategic perspective always inspires me to carry on the research. I also would like to thank my colleagues from the laboratory and others who have provided me their help and suggestions. The author also thank ANSYS Canada and NSERC for the financial support from January 2013 to December 2015.

Dedication

Graduate study is not easy. The constant love and support from my family always gives me the courage to deal with the tough times and the courage to overcome all kinds of difficulties to make this thesis possible during the course of the graduate study.

Chapter 1

Introduction

1.1 Preliminary

The partial differential equations (PDEs) that describe physical phenomena are usually built on the continuous spatial and temporal domains. The PDEs usually consist of two parts: temporal terms and spatial terms. We express the PDEs in the following way:

$$\frac{\partial U}{\partial t} + \frac{\partial f(U)}{\partial x} = 0, \tag{1.1}$$

$$t \geq 0 \qquad x \in \Omega \subseteq R^d, \tag{1.2}$$

$$U(x,t) = u_0 \quad U(\partial\Omega, t) = U_{\partial\Omega}, \tag{1.3}$$

where U is the conservative variables vector, x is the spatial coordinates, $f(U)$ is the flux vector, Ω is the spatial domain, $\partial\Omega$ is the boundary of the spatial domain, R denotes real numbers, and d is the dimension of the problem. If the temporal term $\frac{\partial u}{\partial t}$ is zero, the problem reduces to a steady state problem.

To solve the equations numerically, we must discretize the spatial and temporal domains. The spatial discretization generates a mesh. There are two categories of meshes: one is the structured mesh and the other is the unstructured mesh. The finite difference method, the spectral method, the finite element method, and the finite volume method are the most common ways to represent the spatial terms on the discretized spatial domain. Usually the finite difference method and the spectral method cannot be applied to unstructured meshes while the finite element method and finite volume method can be applied to both mesh types.

After the spatial discretization, a system of ordinary differential equations corresponding to the original partial differential equations (1.1) is generated:

$$\frac{d\bar{U}}{dt} = R(\bar{U}, x). \tag{1.4}$$

The variable $\bar{U}$ in Equation (1.4) is the computational solution vector or discrete solution vector:

$$\bar{U} = \begin{pmatrix} \bar{U}_0 \\ \bar{U}_1 \\ \bar{U}_2 \\ \vdots \\ \bar{U}_{n-1} \end{pmatrix} \tag{1.5}$$

where $\bar{U}_i$ is the component of the discrete solution vector at the ith node or control volume of the mesh. Equation (1.4) is the semi-discretization to the original equation, which is also called the method of lines. Semi-discretization is a good model to isolate the influence of the temporal discretization when studying the spatial discretization. There are many ways to discretize the temporal terms, for instance, the explicit Euler temporal discretization and the implicit Euler temporal discretization. Applying the explicit Euler temporal discretization to Equation (1.4) yields

$$\frac{\bar{U}^{n+1} - \bar{U}^n}{\Delta t} = R\left(\bar{U}^n\right) \tag{1.6}$$

where Δt is the time step. In Equation (1.6) $\bar{U}^n$ is the computational solution at time level n. Suppose $\bar{U}_s^n$ is the exact solution to the fully discretized equation (1.6) at time level n.

Definition 1. The computational error $\Delta\bar{U}^n$ is the difference between the exact solution to the fully discretized equation $\bar{U}_s^n$ and the computational solution $\bar{U}^n$ at the nth time level:

$$\Delta\bar{U}^n = \bar{U}_s^n - \bar{U}^n. \tag{1.7}$$

$\Delta\bar{U}$ refers the computational error in general.

Stability, consistency, and convergence are always the most important properties of numerical schemes. For a well-posed problem, these three properties are related to each other by the Lax Equivalence Theorem:

Theorem 1. *Equivalence Theorem of Lax: For a well-posed initial value problem and a consistent discretization scheme, stability is the necessary and sufficient condition for convergence.*

This statement is quoted directly from Hirsch (2007) [9]. The Lax equivalence theorem, also called the Lax-Richtmyer equivalence theorem, was presented originally by Lax and Ritchtmyer (1956) [12]. A description of this theorem can also be found in the Richtmyer and Morton (1967) [23].

Definition 2. A numerical scheme is *stable* if and only if the norm of the computational solution or the associated computational error remains bounded as the iteration process marches to infinity, i.e.,

$$|\bar{U}^n| \leq C_1 \qquad \text{as n} \to \infty, \tag{1.8}$$

or

$$|\Delta\bar{U}^n| \leq C_2 \qquad \text{as n} \to \infty, \tag{1.9}$$

where C_1 and C_2 are constants independent of n. This is a commonly accepted definition, but it is not very rigorous. For non-normal operators, Condition (1.8) and Condition (1.9) are too strict to be satisfied. To remedy this, a weaker condition that the error grows linearly as the iteration n can be used:

$$|\Delta\bar{U}^n| \leq nC_3, \tag{1.10}$$

where C_3 is independent of n.

1.2 Stability Analysis Methods

Condition (1.8), Condition (1.9), and Condition (1.10) for stability are quite abstract in that one cannot easily determine under what conditions a scheme is stable. To deal with this, there are three main methods to simplify the stability analysis: the von Neumann method, the energy method, and the matrix method. In this subsection, we give a brief introduction to each method.

1.2.1 The Von Neumann Method

The von Neumann method is the most classical method to study the stability property of a numerical scheme. The earliest published description of this method can be seen in Crank and Nicolson (1947) [4] and Charney et al. (1950) [3]. A comprehensive and recent description can be found in Hirsch (2007) [9].

The basic idea of the von Neumann method is to decompose the computational solution or the computational error into a discrete Fourier series. We take the one dimensional advection-diffusion equation as an example:

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = \alpha\frac{\partial^2 u}{\partial^2 x}, \tag{1.11}$$

$$x \in [-\pi, \pi], u(x,0) = u_0, u(-\pi) = u(\pi), \tag{1.12}$$

$$a > 0,\ \alpha > 0. \tag{1.13}$$

The mesh is the one dimensional uniform mesh with $\Delta x = \frac{2\pi}{K}$ which has $K+1$ nodes. For $K = 10$, the mesh is shown in Figure (1.1).

Figure 1.1: 1D uniform mesh with $K = 10$

The first order upwind finite difference for the advection term, and the second order central finite difference for the diffusion term are employed to discretize the spatial terms to yield the semi-discrete equation

$$\frac{d\bar{U}_i}{dt} + a\frac{\bar{U}_{i-1} - \bar{U}_i}{\Delta x} = \alpha\frac{\bar{U}_{i-1} - 2\bar{U}_i + \bar{U}_{i+1}}{\Delta x^2}. \tag{1.14}$$

Explicit Euler temporal discretization is used to discetize temporal terms to obtain the fully discrete equation

$$\frac{\bar{U}_i^{n+1} - \bar{U}_i^n}{\Delta t} + a\frac{\bar{U}_{i-1}^n - \bar{U}_i^n}{\Delta x} = \alpha\frac{\bar{U}_{i-1}^n - 2\bar{U}_i^n + \bar{U}_{i+1}^n}{\Delta x^2}. \tag{1.15}$$

Rearranging Equation (1.15), we have

$$\bar{U}_i^{n+1} = \left(\alpha\frac{\Delta t}{\Delta x^2} - a\frac{\Delta t}{\Delta x}\right)\bar{U}_{i-1}^n + \left(1 + a\frac{\Delta t}{\Delta x} - 2\alpha\frac{\Delta t}{\Delta x^2}\right)\bar{U}_i^n + \alpha\frac{\Delta t}{\Delta x^2}\bar{U}_{i+1}^n. \tag{1.16}$$

The discrete solution $\bar{U}^n$ can be decomposed into a finite Fourier series. For each component, we have

$$\bar{U}_i^n = \sum_{j=0}^{K} V_j^n e^{I\frac{ij\pi}{K}} \tag{1.17}$$

where $I = \sqrt{-1}$. Substitute Equation (1.17) into Equation (1.16) to get

$$\begin{aligned}\sum_{j=0}^{K} V_j^{n+1} e^{I\frac{ij\pi}{K}} &= \left(\alpha\frac{\Delta t}{\Delta x^2} - a\frac{\Delta t}{\Delta x}\right)\sum_{j=0}^{K} V_j^n e^{I\frac{(i-1)j\pi}{K}} \\ &+ \left(1 + a\frac{\Delta t}{\Delta x} - 2\alpha\frac{\Delta t}{\Delta x^2}\right)\sum_{j=0}^{K} V_j^n e^{I\frac{ij\pi}{K}} \\ &+ \alpha\frac{\Delta t}{\Delta x^2}\sum_{j=0}^{K} V_j^n e^{I\frac{(i+1)j\pi}{K}}. \end{aligned} \tag{1.18}$$

For the jth harmonic, we have

$$V_j^{n+1} = \left(\alpha\frac{\Delta t}{\Delta x^2} - a\frac{\Delta t}{\Delta x}\right) V_j^n e^{I\frac{-j\pi}{K}} + \left(1 + a\frac{\Delta t}{\Delta x} - 2\alpha\frac{\Delta t}{\Delta x^2}\right) V_j^n + \alpha\frac{\Delta t}{\Delta x^2} V_j^n e^{I\frac{j\pi}{K}}. \tag{1.19}$$

We define the amplification factor

$$G_j \triangleq \frac{V_j^{n+1}}{V_j^n}. \tag{1.20}$$

To be stable, the following condition needs to be satisfied:

$$\left|G_j\right| \leq 1 \quad \text{for all j} = 0, \ldots \text{K}. \tag{1.21}$$

This condition is the von Neumann stability condition (Hirsch (2007) [9]). Combining Equation (1.19) and Equation (1.20) gives

$$G_j = \left(\alpha\frac{\Delta t}{\Delta x^2} - a\frac{\Delta t}{\Delta x}\right) e^{I\frac{-j\pi}{K}} + \left(1 + a\frac{\Delta t}{\Delta x} - 2\alpha\frac{\Delta t}{\Delta x^2}\right) + \alpha\frac{\Delta t}{\Delta x^2} e^{I\frac{j\pi}{K}}. \tag{1.22}$$

By Euler's formula, we have:

$$\begin{aligned} G_j &= \left(\alpha\frac{\Delta t}{\Delta x^2} - a\frac{\Delta t}{\Delta x}\right)\cos\left(\frac{j\pi}{K}\right) - I\left(\alpha\frac{\Delta t}{\Delta x^2} - a\frac{\Delta t}{\Delta x}\right)\sin\left(\frac{j\pi}{K}\right) \\ &+ \left(1 + a\frac{\Delta t}{\Delta x} - 2\alpha\frac{\Delta t}{\Delta x^2}\right) + \alpha\frac{\Delta t}{\Delta x^2}\cos\left(\frac{j\pi}{K}\right) + I\alpha\frac{\Delta t}{\Delta x^2}\sin\left(\frac{j\pi}{K}\right) \\ &= \left(1 + a\frac{\Delta t}{\Delta x} - 2\alpha\frac{\Delta t}{\Delta x^2}\right) + \left(2\alpha\frac{\Delta t}{\Delta x^2} - a\frac{\Delta t}{\Delta x}\right)\cos\left(\frac{j\pi}{K}\right) \\ &+ Ia\frac{\Delta t}{\Delta x}\sin\left(\frac{j\pi}{K}\right). \end{aligned}$$

To be stable, $|G| \leq 1$, i.e.,

$$\left(\left(1+a\frac{\Delta t}{\Delta x}-2\alpha\frac{\Delta t}{\Delta x^2}\right)+\left(2\alpha\frac{\Delta t}{\Delta x^2}-a\frac{\Delta t}{\Delta x}\right)\cos\left(\frac{j\pi}{K}\right)\right)^2+\left(a\frac{\Delta t}{\Delta x}\sin\left(\frac{j\pi}{K}\right)\right)^2 \leq 1 \quad \text{for j} = 0,\ldots \text{K}. \tag{1.23}$$

1.2.2 The Energy Method

The energy method has a long history of being used in the stability analysis of partial differential equations. An early example of the energy method's application to numerical methods can be found in Richtmyer and Morton (1967) [23]. The energy method usually defines a norm of the computational solution as energy and provides the condition under which the energy monotonically decreases, therefore resulting in a stable system.

For linear problems and nonlinear problems like the Euler equations that are homogeneous of degree one, the term $R(\bar{U})$ in Equation (1.4) can be rewritten as

$$R(\bar{U}) = \frac{\partial R}{\partial \bar{U}}\bar{U}. \tag{1.24}$$

Substitute Equation (1.24) into Equation (1.4) to give

$$\frac{d\bar{U}}{dt} = \frac{\partial R}{\partial \bar{U}}\bar{U}. \tag{1.25}$$

We define the standard norm of the computational solution as energy:

$$E = \bar{U}^{\intercal}\bar{U}. \tag{1.26}$$

Multiply both sides of Equation (1.25) by $U^{\intercal}$ to give

$$\bar{U}^{\intercal}\frac{d\bar{U}}{dt} = \frac{1}{2}\frac{dE}{dt} = \bar{U}^{\intercal}\frac{\partial R}{\partial \bar{U}}\bar{U}. \tag{1.27}$$

To be stable, the energy E should not increase as time t, i.e., $\frac{dE}{dt} \leq 0$. As a result, we need

$$\bar{U}^{\intercal}\frac{\partial R}{\partial \bar{U}}\bar{U} \leq 0. \tag{1.28}$$

This condition is sufficient for a normal operator. For an ill-conditioned non-normal operator, this condition is not sufficient. D. Levy and E. Tadmor (1998) [14] proposed a strengthened condition

$$\bar{U}^{\intercal}\frac{\partial R}{\partial \bar{U}}\bar{U} \leq -\eta \left|\frac{\partial R}{\partial \bar{U}}\bar{U}\right|. \tag{1.29}$$

Lenferink and Spijker (1991) [13] came up with another approach to deal with ill-conditioned non-normal operators.

1.2.3 The Matrix Method

In general, for a linear problem, the evolution of the computational error between two successive iterations can be mapped by the following relation:

$$\Delta\bar{U}^{n+1} = M\Delta\bar{U}^{n} + B, \tag{1.30}$$

where M is a matrix with size equal to the number of unknowns, and B is a vector associated with boundary conditions. For some boundary conditions and boundary condition enforcement methods, B is zero. We do not take B into consideration here. Suppose the initial computational error is $\Delta\bar{U}^{0}$. For a linear problem, M is constant; therefore, $\Delta\bar{U}^{n} = M^{n}\Delta\bar{U}^{0}$. To be stable, the norm of the computational error $\Delta\bar{U}^{n}$ needs to be bounded as n becomes infinite:

$$|\Delta\bar{U}^{n}| = \left|M^{n}\Delta\bar{U}^{0}\right| \leq C \quad \text{as n} \to \infty.$$

we also have:

$$|\Delta\bar{U}^{n}| = \left|M^{n}\Delta\bar{U}^{0}\right| \leq |M^{n}|\left|\Delta\bar{U}^{0}\right|.$$

we can see that the behavior of $\Delta\bar{U}$ is determined by M. To have $|\Delta\bar{U}^{n}|$ be bounded, the following condition needs to be satisfied (see e.g., Horn and Johnson (1990) [10]):

1. The largest modulus of the eigenvalues of M should not be larger than one.
2. If the modulus equals one, then the associated eigenvalues must be semisimple.

For a steady flow problem, the solution evolution starts from a guessed solution, which does not satisfy the discretized equations. To be able to converge to the solution of the discretized equations, the computational error needs to be machine zero as n become sufficiently large. Therefore, we need the largest modulus of the eigenvalues to be smaller than one.

A more detailed description of this method can be found in Hirsch (2007) [9].

For an ill-conditioned operator, though this condition is satisfied, there will be a large transient growth before the eventual decay. This phenomenon is well

documented in Reddy and Trefethen (1992) [22], in which an approach to deal with the non-normality is presented.

For a nonlinear problem, if a linearization is used, we can get the a mapping for the computational error of two successive iteration with the same form as Equation (1.30).

1.2.4 Remarks

Among these three methods, the von Neumann method is the most convenient one, providing information about how various factors affect the stability and accuracy properties of a numerical scheme. One drawback of the von Neumann method is that it can only be applied to linear problems with periodic boundary conditions on uniform structured meshes. However, the von Neumann method can provide valuable guidelines on stability and accuracy of more general structured mesh cases as well as including nonlinear problems. In general, the von Neumann method cannot be applied to unstructured discretizations.

The energy method can be applied both to structured discretizations and unstructured discretizations. By the energy method, one may figure out general information about the stability properties of a scheme. One drawback of the energy method is that its success relies on a suitable form of the defined energy, i.e., a suitable norm of the computational error or solution. (see, e.g., Giles (1997) [6] and F. Haider et al. (2009) [7]). In addition, the analysis associated with the energy method is usually in the manner of functional analysis, which is not easy. It is also hard to take boundary conditions into consideration for the energy method. For high order schemes, the analysis is much more complicated than the first and second order schemes.

Both for the von Neumann method and the energy method, one must study the structure of the system to perform the analysis. However, for the matrix method, one just needs to calculate the eigenvalues and in general one does not need to study the structure of the system. The matrix method can easily incorporate boundary conditions. One drawback of the matrix method is that the eigenvalue calculation is in general expensive. Also, it usually does not provide some general guidelines, like how mesh type, reconstruction, the coordinates of the mesh, etc., affect the stability.

For a numerical scheme on a structured mesh for a linear problem with periodic condition, the corresponding matrix is a circulant matrix, and the eigenvalues can be calculated analytically. For this case, the matrix method is equivalent to the von Neumann method, suggesting that the von Neumann method is a special case of the matrix method. Compared with a normal matrix, a non-normal matrix is much more difficult to deal with. For further information, please see Reddy and

Trefethen (1992) [22], Giles (1997) [6], and the references therein.

The problem of interest in this thesis is the steady compressible inviscid flow problem, and we employ a high order (third or fourth) finite volume method to discretize the spatial terms on unstructured meshes. The Jacobian matrix of the spatial discretization therefore is non-symmetric and the condition number is usually large. However, in practice, we do not encounter large transient growth before the eventual decay, and for a steady problem, the intermediate solution during time-marching is not of central concern. Though the condition number is large, the eigenvalue calculation is still reliable. Based on these considerations and observations, we use the matrix method in this thesis to implement stability analysis and stabilization due to its convenience.

1.3 Literature Review

By using stability analysis, we can know if a scheme is stable or not, and for a full discretization, we can find the maximum stable time step. It is also quite valuable to study how various factors — for instance, the reconstruction method, the mesh type, etc. — affect the stability properties of a numerical scheme. In this section, we give a brief review of current literature on the stability analysis of the finite volume method on unstructured meshes. Please be aware that there are not many articles on the stability analysis of the unstructured finite volume method.

M. B. Giles (1987) [5] developed a framework of performing energy stability analysis for unstructured discretizations. In this article, the author used the 1D convection equation as the model problem. The basic ideas are: first present the properties that suffice for stability for the PDEs, the semi-discretization, and the full-discretization respectively; if the corresponding properties are satisfied, the PDEs, the semi-discretization, and the full discretization are stable automatically. Then prove that the PDEs of interest and semi-discretization satisfy the designed properties under some assumptions, therefore being stable; for the full-discretization to satisfy the properties, the time-step limit can then be obtained for multi-stage explicit time-stepping. The author extended the method to first order systems of equations and the Euler equations. It is supposed to give more proper time-steps than classical von Neumann method for the unstructured discretization employing multi-stage time-stepping. However, the analysis in this article is not verified by numerical results.

M. B. Giles (1997) [6] developed a method to perform time-step stability analysis for unstructured discretizations of the Navier-Stokes equations. In this article, the energy method is used to obtain the maximum time-step for stability. To deal with the large transient growth, the algebraic stability condition defined in

Lenferink and Spijker (1991) [13] is employed. The basic ideas of the approach is: first, assume the flow is a uniform flow with periodic boundary conditions; second, prove this flow is stable under small perturbation; third, prove the semi-discretization is stable as well; fourth, to have a stable full-discretization, the time-step is obtained to satisfy the algebra stability condition. In this article, the author obtained the time-step limits both for global and local time-steps for Runge-Kutta time-stepping for a Galerkin discretization of the Navier-Stokes equations on unstructured meshes.

P. Moinier and M. B. Giles (2002) [18] implemented stability analysis for the preconditioned discrete Euler equations with Runge-Kutta time-stepping on unstructured meshes by the energy method based on the algebraic stability condition defined in Lenferink and Spijker (1991) [13]. The approach is similar to the approach in M. B. Giles (1997) [6] . Compared with M. B. Giles (1997) [6], Jacobi preconditioning and low Mach preconditioning are included in the analysis. In addition, solid wall boundary conditions are also taken into consideration. Moinier and Giles implemented numerical tests to show that the presented method can give a time-step limit close to the actual time-step limit. However, for some cases, the time-step limit are too large, which may be because the flow has strong shock-wave, violating the uniform flow assumption.

In the analysis in M. B. Giles (1997) [6] and P. Moinier and M. B. Giles (2002) [18], uniform flow and periodic boundary conditions are assumed. In practice, these conditions are usually not satisfied. Therefore, the results given by this method might not be correct, which is seen in P. Moinier and M. B. Giles (2002). It is also complicated to extend the energy method to higher order schemes and take the boundary conditions into consideration.

By using the energy method and combining it with the matrix method, F. Haider et al. (2009) [7] studied the stability of the MUSCL ([26, 27]) discretization of the linear advection equation on general unstructured meshes and investigated the influence of the slope reconstruction method, the mesh type, and the stencil size on the stability. The authors proposed a new approximate and qualitative criterion to measure how the piecewise linear slope reconstruction affects the stability of the MUSCL scheme, for two reasons: even if the scheme is stable, the defined energy can locally increase when using the energy method and the relations between the slope reconstruction and the eigenvalues are too complicated to analyze for unstructured discretizations when using the matrix method. The investigation about stability under this new criterion gives two important conclusions: (1) least squares slope reconstruction is the most stable method; (2) extending the reconstruction stencil can increase the stability of a scheme, which is valid at least for the least squares slope reconstruction scheme. Extensive numerical tests were implemented and verified the two conclusions. The energy stability analysis method

in this article studies the stability by studying the structure of the reconstruction in the manner of functional analysis. The conclusions drawn by this approach can be quite general; however, the analysis is quite hard. It would be harder if this approach were extended to high order schemes. Also, the success of the energy method analysis relies on the suitable definition of energy, which led the authors to seek a new criterion.

The current literature contains no efforts to optimize the spatial discretization in a quantitative and controllable way, which is one of the topics of this thesis.

1.4 Motivation and Outline

Structured meshes have obvious efficiency and accuracy advantage, but because of its flexibility and automatic generation, the unstructured mesh is commonly used in simulations on complex geometry domains. However, in practice failing to converge to the solution happens often, and a long trial and error process is sometimes required to generate a suitable mesh to reach the converged solution. An approach to optimize the mesh for stability would be quite valuable. As mentioned previously, currently the study on the unstructured finite volume method is far from being sufficient. It is worthy to develop a model to study the stability properties of the unstructured finite volume method and a model to stabilize the unstable numerical schemes.

In this thesis, we develop a model based on the matrix method to implement stability analysis for the discretized Euler equations on unstructured meshes. We also develop a method to optimize the mesh to stabilize unstable spatial discretization cases.

The reminder of this thesis is organized in five chapters. Chapter 2 presents a model to implement stability analysis. Chapter 3 verifies the stability analysis model. Chapter 4 presents the method to stabilize the unstable spatial discretization at the fixed point. Chapter 5 presents the results of the stabilization at a non-fixed point. Chapter 6 summaries the study in this thesis and presents a prospective view of future work.

Chapter 2

Essential Background and Stability Analysis

2.1 Governing Equations and Numerical Methods

The physical problems of interest in this thesis are steady compressible flow problems, for which the governing equations are the Euler equations for inviscid flows and the Navier-Stokes equations for viscous flows. As a good starting problem, only the two dimensional Euler equations are studied in this thesis, for the presented methods can be easily extended to the two dimensional Navier-Stokes equations, the three dimensional Euler equations, and the three dimensional Navier-Stokes equations. In this thesis, the Euler equations are solved numerically by using a finite volume scheme employing least squares solution reconstruction [19] and Roe flux difference splitting [24] on unstructured meshes. In this section, a simple introduction is given that is sufficient for expressing the new ideas presented in this thesis.

2.1.1 Physical Governing Equations

The Euler equations describe the motion of compressible inviscid flows. For the convenience of applying the finite volume method, here the conservation form is given:

$$\frac{d}{dt}\int_{V_i} U dV + \oint_{S_i} F dS = 0, \tag{2.1}$$

$$U = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix} \quad F = \begin{bmatrix} \rho u_n \\ \rho u u_n + P\hat{n}_x \\ \rho v u_n + P\hat{n}_y \\ (E+P)u_n \end{bmatrix}, \tag{2.2}$$

where U is the conservative variables vector, V is the control volume, S is the surface of the control volume, and F is the flux vector. In Equation (2.2), ρ, u, v, and E are density, velocity in the x-direction, velocity in the y-direction, and

total energy respectively. Suppose $\hat{n}$ is the outward unit normal vector. $\hat{n}_x$ is the projection of $\hat{n}$ on the x-direction and $\hat{n}_y$ is the projection of $\hat{n}$ on the y-direction. In Equation (2.2), $u_n = u\hat{n}_x + v\hat{n}_y$. The pressure P is related to the conservative variables by the ideal gas equation of state:

$$P = (\gamma - 1)\left(E - \frac{1}{2}\rho\left(u^2 + v^2\right)\right). \tag{2.3}$$

2.1.2 Numerical Discretization For Spatial Terms

In this thesis, we consider an unstructured finite volume scheme employing least squares solution reconstruction and Roe flux difference splitting [24]. The spatial discretization generates a nonlinear real value vector function as following:

$$\oint_{S_i} F dS = R(\bar{U}) \tag{2.4}$$

where the term $R(\bar{U})$ is the residual term and $\bar{U}$ is the solution vector or computational solution vector. Detailed information on the spatial discretization can be found in [19]. A numerical method to solve the steady Euler equations is aimed at seeking a solution vector $\bar{U}_s$ satisfying

$$R(\bar{U}_s) = 0 \tag{2.5}$$

The residual term $R(\bar{U})$ can be interpreted as a function of the coordinates of vertices of the mesh if the mesh's influence on the system is taken into consideration, which is of central interest to this thesis:

$$\oint_{S_i} F dS = R(\bar{U}) = R(\bar{U}, \zeta) \tag{2.6}$$

where ζ denotes the vector of the coordinates of vertices of the mesh discretizing the spatial domain.

2.1.3 Temporal Discretization and Time Stepping Methods

Since the problems of interest are steady problems and we expect only one solution to Equation (2.5), the accuracy of the temporal discretization only affects the transient procedure to reach the converged solution but does not influence the solution accuracy. Among many temporal discretization methods, only the backward Euler method is studied in this thesis.

Backward Euler Time Stepping Method

The governing equations discretized in space can be written as

$$\frac{d\bar{U}}{dt} = -R(\bar{U}). \tag{2.7}$$

By applying the backward Euler method to the temporal terms in Equation (2.7), we have

$$\frac{\bar{U}^k - \bar{U}^{k-1}}{\Delta t^k} = -R\left(\bar{U}^k\right), \tag{2.8}$$

where k is the iteration count, meaning the kth time level, and Δt is the time-step. The term on the right-hand side of Equation (2.8) can be expanded by a Taylor series in the following way:

$$R\left(\bar{U}^k\right) = R\left(\bar{U}^{k-1}\right) + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}^{k-1}} \left(\bar{U}^k - \bar{U}^{k-1}\right) + O\left(\left(\bar{U}^k - \bar{U}^{k-1}\right)^2\right) \tag{2.9}$$

where the term $\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}^{k-1}}$ is the Jacobian matrix. By combining Equation (2.8) and Equation (2.9), and dropping the high order terms, we get the following equation:

$$\frac{\bar{U}^k - \bar{U}^{k-1}}{\Delta t^k} = -\left(R\left(\bar{U}^{k-1}\right) + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}^{k-1}} \left(\bar{U}^k - \bar{U}^{k-1}\right)\right) \tag{2.10}$$

where the term $\bar{U}^k - \bar{U}^{k-1}$ can be denoted by a variable called the solution update $\delta\bar{U}$. As a result, we obtain a concise form:

$$\left(\frac{I}{\Delta t^k} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}^{k-1}}\right) \delta\bar{U}^k = -R\left(\bar{U}^{k-1}\right) \tag{2.11}$$

where I is the identity matrix. The solution update at the kth iteration $\delta\bar{U}^k$ can be obtained by solving this linear equation. The solution at the kth iteration $\bar{U}^k$ is updated in the following way:

$$\bar{U}^k = \bar{U}^{k-1} + \alpha\delta\bar{U}^k \tag{2.12}$$

where α is a positive factor, which equals one for the backward Euler method. $\alpha > 1$ corresponds to overrelaxiation, and $\alpha < 1$ to underrelaxation.

2.1.4 Summary of Numerical Methods

Here we summarize the whole procedure for applying a numerical scheme to solve the two dimensional Euler equations:

(1) Discretize the spatial terms and discretize the temporal terms with the backward Euler method:

$$\left(\frac{I}{\Delta t^k} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}^{k-1}}\right)\delta\bar{U}^k = -R\left(\bar{U}^{k-1}\right). \tag{2.13}$$

k starts from $k=1$ with an initial guess: $\bar{U}=\bar{U}^0$. We get the solution update $\delta\bar{U}^k$ by solving Equation (2.13).

(2) Update the solution by $\delta\bar{U}^k$:

$$\bar{U}^k = \bar{U}^{k-1} + \alpha\delta\bar{U}^k. \tag{2.14}$$

If the computational solution satisfies the convergence criterion, for instance, the L_2 norm of $R(\bar{U})$ is smaller than 10^{-10}, then stop here; otherwise, go to (1).

Hence, the mapping of the computational solutions of two successive iterations is

$$\bar{U}^k = \bar{U}^{k-1} + \left(\frac{I}{\Delta t^k} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}^{k-1}}\right)^{-1}\left(-R\left(\bar{U}^{k-1}\right)\right). \tag{2.15}$$

Therefore, the mapping of the computational solution of the kth iteration and initial guess is

$$\bar{U}^k = f\left(f\left(f^{\cdots}\left(f\left(\bar{U}^0\right)\right)\right)\right) \quad k\text{th level } f, \tag{2.16}$$

where we apply k iterations of the nonlinear function

$$f\left(\bar{U}^{j+1}\right) = \bar{U}^j + \left(\frac{I}{\Delta t^j} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}^j}\right)^{-1}\left(-R\left(\bar{U}^j\right)\right). \tag{2.17}$$

We can also use a simpler form to denote Equation (2.16):

$$\bar{U}^k = f^k\left(\bar{U}^0\right). \tag{2.18}$$

The central focus of this thesis is studying the stability of the solution procedure.

2.2 Stability Analysis Methodology

For the Euler equations, Mapping (2.18) is nonlinear. The inherent nonlinearity of the Euler equations makes analysis much more complicated than a linear system. In general, for a nonlinear system, analysis around the converged solution (the fixed point) is applied to study the system. Instead of focusing on the computational solution, a typical fixed point analysis regards the computational error $\Delta\bar{U}$ as the object to study. If the computational error converges to 0 as k approaches infinity, the system is stable; otherwise it is unstable. This simplifies the analysis.

In this section, we present in detail the model originally developed in this thesis to implement the stability analysis for the discretized systems of physical equations. The main ideas were obtained from communication in October 2014 with Dr. Christopher Hill at ANSYS Corporation [8]. The original methodology proposed by C. Hill is aimed at the stability analysis and stabilization for the dynamical system coupling the spatial discretization with the temporal discretization, which includes not only simple temporal discretization methods like the backward Euler time stepping method, but also sophisticated temporal discretization methods. Besides presenting the originally proposed model, this thesis focuses on the backward Euler time stepping method and constructs a stability condition solely for the spatial discretization. One of the remarkable points of the developed model is that instead of employing the conventional way of implementing the fixed point analysis, a simpler approach is developed that drops the high order terms.

2.2.1 Mathematical Model to Implement Stability Analysis at Fixed Point

As the usual way of implementing fixed point analysis, instead of studying the Mapping (2.16) relating the initial computational solution $\bar{U}^0$ and the computational solution of the kth iteration $\bar{U}^k$, we focus on the behavior of the computational error. We start from the mapping of the computational solution between two successive iterations and then shift to the mapping of the computational error. Recall Equation (2.13) and Equation (2.12):

$$\left(\frac{I}{\Delta t^k} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}^{k-1}}\right)\delta\bar{U}^k = -R\left(\bar{U}^{k-1}\right), \tag{2.19}$$

$$\bar{U}^k = \bar{U}^{k-1} + \alpha\delta\bar{U}^k. \tag{2.20}$$

Suppose for a certain k, $\bar{U}^{k-1}$ is the converged solution of Equation (2.5) and the fixed point of time-stepping defined by Equation (2.19) and Equation (2.20); there-

fore, we have $R\left(\bar{U}^{k-1}\right)=0$, $\delta\bar{U}^k=0$, and $\bar{U}^k\equiv\bar{U}^{k-1}$.[1] A stable system should be able to recover to the converged solution after a small perturbation. Following this idea, to investigate stability, we perturb the solution randomly — therefore the perturbation is rich in directions — and apply the perturbed solution into the system to investigate its effects on the time-stepping process:

$$\bar{U}_p=\bar{U}_s+\Delta\bar{U}_p \tag{2.21}$$

where $\Delta\bar{U}_p$ is the perturbation, $\bar{U}_s$ is the steady solution, and $\bar{U}_p$ is the computational solution after perturbation. According to the definition of the computational error defined in Definition 1, $\Delta\bar{U}_p$ is also the computational error $\Delta\bar{U}$. To study the stability of the time-stepping process, we insert the perturbed solution into the time-stepping process. After k iterations, we can relate the computational solution to the converged solution at the kth iteration:

$$\bar{U}_p^k=\bar{U}_s+\Delta\bar{U}^k. \tag{2.22}$$

By the definition of the vector norm, we have:

$$\left|\bar{U}_p^k\right|=\left|\bar{U}_s+\Delta\bar{U}^{k-1}\right|\leq\left|\bar{U}_s\right|+\left|\Delta\bar{U}^{k-1}\right| \tag{2.23}$$

where the norm of the fixed point solution $\bar{U}_s$ is fixed, which implies that the behavior of the norm of the computational solution $\bar{U}_p$ inherits from the computational error $\Delta\bar{U}$. As a result, we can just focus on the asymptotic behavior of the computational error $\Delta\bar{U}$. Insert Equation (2.22) into Equation (2.19), and we get:

$$\begin{gathered}\left(\frac{I}{\Delta t^{k+1}}+\frac{\partial\bar{R}}{\partial\bar{U}}^k\bigg|_{\bar{U}=\bar{U}_p^k}\right)\delta\bar{U}^{k+1}=-R\left(\bar{U}_p^k\right),\\ \Downarrow\\ \left(\frac{I}{\Delta t^{k+1}}+\frac{\partial\bar{R}}{\partial\bar{U}}^k\bigg|_{\bar{U}=\bar{U}_s+\Delta\bar{U}^k}\right)\delta\bar{U}^{k+1}=-R\left(\bar{U}_s+\Delta\bar{U}^k\right).\end{gathered} \tag{2.24}$$

The Jacobian matrix on the left-hand side is calculated based on the computational solution $\bar{U}_s+\Delta\bar{U}^k$. To reduce the complexity of analysis, this Jacobian matrix is replaced by the Jacobian matrix calculated based on the fixed point solution. The right-hand terms can be linearized by the Taylor expansion as:

[1] In the context of scientific computation, there is no real " 0 " . Here " 0 " is a vector or scalar as close to 0 as can be reached in the system, constrained by the precision of data and the system itself. Therefore, we can say after a certain k, $\bar{U}^k$ is the fixed point since the changes in $\bar{U}^k$ are meaningless afterward.

$$-R\left(\bar{U}_s+\Delta\bar{U}^k\right)=-\left(R\left(\bar{U}_s\right)+\left.\frac{\partial\bar{R}}{\partial\bar{U}}\right|_{\bar{U}=\bar{U}_s}\left(\Delta\bar{U}^k\right)\right)+O\left(\Delta\bar{U}^k\right)^2 \tag{2.25}$$

where $R\left(\bar{U}_s\right)=0$. Combining these, an equation with a more clear form can be generated after some simple algebraic operations:

$$\left(\frac{I}{\Delta t^{k+1}}+\left.\frac{\partial\bar{R}}{\partial\bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k+1}=-\left(\left.\frac{\partial\bar{R}}{\partial\bar{U}}\right|_{\bar{U}=\bar{U}_s}\left(\Delta\bar{U}^k\right)\right)+O\left(\Delta\bar{U}^k\right)^2. \tag{2.26}$$

Since we aim at studying the asymptotic behavior of the computational error, and there is another variable $\delta\bar{U}^{k+1}$ in the equation, we need to express the variable solution update $\delta\bar{U}^{k+1}$ in terms of the variable $\Delta\bar{U}^k$. Retrieve equation (2.12) and equation (2.22):

$$\bar{U}^k=\bar{U}^{k-1}+\alpha\delta\bar{U}^k, \tag{2.27}$$

$$\bar{U}_p^k=\bar{U}_s+\Delta\bar{U}^k. \tag{2.28}$$

Combine these two equations, and we get:

$$\frac{\Delta\bar{U}^k-\Delta\bar{U}^{k-1}}{\alpha}=\delta\bar{U}^k. \tag{2.29}$$

By using Equation (2.29) in Equation (2.26), an equation relating the computational errors of two successive iterations can be derived :

$$\left(\frac{I}{\Delta t^{k+1}}+\left.\frac{\partial\bar{R}}{\partial\bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\Delta\bar{U}^{k+1}=\left(\frac{I}{\Delta t^{k+1}}+(1-\alpha)\left.\frac{\partial\bar{R}}{\partial\bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\Delta\bar{U}^k+O\left(\Delta\bar{U}^k\right)^2. \tag{2.30}$$

Drop the higher order term $O\left(\Delta\bar{U}^k\right)^2$, replace $k+1$ by k , and we can get a concise formula :

$$\left(\frac{I}{\Delta t^k}+\left.\frac{\partial\bar{R}}{\partial\bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\Delta\bar{U}^k=\left(\frac{I}{\Delta t^k}+(1-\alpha)\left.\frac{\partial\bar{R}}{\partial\bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\Delta\bar{U}^{k-1}. \tag{2.31}$$

If the backward Euler time stepping method is applied, the time step Δt is constant and $\alpha=1$, resulting in a simpler form of Equation (2.31) as following:

$$\left(\frac{I}{\Delta t}+\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\Delta\bar{U}^k=\frac{1}{\Delta t}\Delta\bar{U}^{k-1}. \tag{2.32}$$

We can easily see that System (2.31) is linear, which simplifies the analysis of the system dramatically, and as shown by the data in Chapter 3, this linearity is valid in the neighbor around the converged solution. Assume the matrix on the left-hand side of equation (2.31) is not singular, which is valid for the problems of interest in this thesis, then we can have a more direct mapping of $\Delta\bar{U}$ of two successive iterations:

$$\Delta\bar{U}^k=\left(\frac{I}{\Delta t^k}+\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)^{-1}\left(\frac{I}{\Delta t^k}+(1-\alpha)\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\Delta\bar{U}^{k-1}. \tag{2.33}$$

With the help of the recurrence relation of $\Delta\bar{U}$ of Equation (2.33), the mapping of the initial computational error $\Delta\bar{U}^0$ and the computational error at the kth iteration $\Delta\bar{U}^k$ can be deduced as following:

$$\Delta\bar{U}^k=\left(\left(\frac{I}{\Delta t^k}+\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)^{-1}\left(\frac{I}{\Delta t^k}+(1-\alpha)\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\right)^k\Delta\bar{U}^0. \tag{2.34}$$

Since the term $\left(\frac{I}{\Delta t^k}+\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)^{-1}\left(\frac{I}{\Delta t^k}+(1-\alpha)\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)$ is a constant matrix, we use a letter A to denote it to obtain a concise formulation of Equation (2.31):

$$\Delta\bar{U}^k=A^k\Delta\bar{U}^0. \tag{2.35}$$

We get an approximate mapping relating the initial computational error $\Delta\bar{U}^0$ and the computational error at the kth iteration $\Delta\bar{U}^k$. In fact, for the solution update $\Delta\bar{U}$ and flux integral $R(\bar{U})$, we can also derive mappings similar to Equation (2.35), the details of which can be seen in Appendix A. Naturally, the next question is under what circumstances the computational error converges to 0, and therefore the system is stable; and under what circumstances it diverges.

2.2.2 Eigenanalysis of the Model

A common way to study the behavior of Mapping (2.35) is spectral analysis. In this subsection, we discuss in detail how we employ spectral analysis to derive the stability conditions.

Without loss of generality, suppose matrix A is a $n \times n$ square matrix with a set of eigenvalues $(\lambda_0, \lambda_1, \lambda_2, \ldots, \lambda_{n-1})$, a set of right eigenvectors $(x_0, x_1, x_2, \ldots, x_{n-1})$ and a set of left eigenvectors $\left(y_0^*, y_1^*, y_2^*, \ldots, y_{n-1}^*\right)$, satisfying the following equations by definition:

$$Ax_i = \lambda_i x_i \quad i = 0, 1, 2, \ldots, n-1, \tag{2.36}$$

$$y_i A = \lambda_i y_i \quad i = 0, 1, 2, \ldots, n-1. \tag{2.37}$$

Suppose there are no repeated eigenvalues for matrix A, which always holds for the problems of interest in this thesis — the discretized equations inherited from the steady 2D Euler equations. Since the right eigenvectors $(x_0, x_1, x_2, \ldots, x_{n-1})$ are linearly independent, they can span a vector space of dimension n. Similar to the right eigenvectors, the set of left eigenvectors $\left(y_0^*, y_1^*, y_2^*, \ldots, y_{n-1}^*\right)$ can span a vector space of dimension n as well. Therefore, we can decompose ΔU by the right eigenvectors in the following way:

$$\Delta \bar{U} = \sum_{i=0}^{i=n-1} a_i x_i \tag{2.38}$$

where a_i are the eigendecomposition coefficients, calculated with the help of the left eigenvectors:

$$y_k^* \Delta \bar{U} = y_k^* \left(\sum_{i=0}^{i=n-1} a_i x_i \right) \quad k = 0, 1, 2, \ldots, n-1. \tag{2.39}$$

Applying the orthogonality of the left and the right eigenvectors:

$$y_k^* x_i = \begin{cases} 0 & k \neq i \\ C_4 & k = i \end{cases} \tag{2.40}$$

where C_4 is a scalar not equal to zero, we get

$$a_i = \frac{y_i^* \Delta \bar{U}}{y_i^* x_i} \quad i = 0, 1, 2, \ldots, n-1. \tag{2.41}$$

Now, we study the behavior of ΔU under Mapping (2.35) in the eigendecomposition. Multiply Equation (2.38) by the matrix A from the left:

$$A \Delta \bar{U} = \sum_{i=0}^{i=n-1} a_i A x_i. \tag{2.42}$$

By applying Equation (2.36), we get

$$A\Delta\bar{U} = \sum_{i=0}^{i=n-1} a_i A x_i = \sum_{i=0}^{i=n-1} a_i \lambda_i x_i. \tag{2.43}$$

Hence, after multiplying k times as in Equation (2.35), We get:

$$A^k\Delta\bar{U} = \sum_{i=0}^{i=n-1} a_i A x_i = \sum_{i=0}^{i=n-1} a_i \lambda_i^k x_i \tag{2.44}$$

Therefore, the computational error at the kth iteration $\Delta\bar{U}^k$ can be mapped to the initial computational error ΔU^0 as :

$$\Delta\bar{U}^k = A^k\Delta\bar{U}^0 = \sum_{i=0}^{i=n-1} a_i^0 \lambda_i^k x_i \tag{2.45}$$

where a_i^0 are the eigendecomposition coefficients associated with the initial computational error $\Delta\bar{U}^0$. If not explicitly stated, a_i^0 is denoted by a_i hereafter.

Next, we will show the behavior of the computational error under Mapping (2.35) is dominated by the largest eigenvalues in norm and the associated eigenvectors. From now on, in this thesis, largest eigenvalue(s) means the largest eigenvalue(s) in norm if not specifically stated. Given a previous assumption that there are no repeated eigenvalues for matrix A, the largest eigenvalues are either a singleton real eigenvalue or a conjugate complex pair. Suppose the eigenvalues are sorted in the descending order of magnitude with respect to the count i. Because complex eigenvalues are common for the problems of interest, we use Euler's formula to present eigenvalues in a convenient form:

$$\lambda_i = r_i e^{I\theta_i} \tag{2.46}$$

where r_i is the norm of eigenvalue λ_i; I is the imaginary unit so that $I = \sqrt{-1}$; and θ_i is the associated angle in the complex plane. We analyze the behaviors of the mappings by sorting them into two categories in terms of the characteristics of the largest eigenvalues.

(1) We suppose the largest eigenvalue of the matrix A is a singleton real scalar. Rewrite Equation (2.45) and separate the largest eigenvalue from the rest:

$$\Delta\bar{U}^k = A^k\Delta\bar{U} = \sum_{i=0}^{i=n-1} a_i \lambda_i^k x_i = \lambda_0^k \sum_{i=0}^{i=n-1} a_i \left(\frac{\lambda_i}{\lambda_0}\right)^k x_i \tag{2.47}$$

By applying Equation (2.46), we have:

$$\Delta\bar{U}^k = r_0^k \sum_{i=0}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i. \tag{2.48}$$

To study the behavior of the computational error $\Delta\bar{U}^k$, an asymptotic analysis is implemented. For $\left|\Delta\bar{U}^k\right|$, we have

$$\begin{aligned}
\left|\Delta\bar{U}^k\right| &= \left| r_0^k \sum_{i=0}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i \right| \\
&= \sqrt{\left(\left(r_0^k \left(\sum_{i=0}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right)\right)^* r_0^k \left(\sum_{i=0}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right)\right)} \\
&= \sqrt{\left(\left(r_0^k \left(a_0 x_0 + \sum_{i=1}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right)\right)^* r_0^k \left(a_0 x_0 + \sum_{i=1}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right)\right)} \\
&= \sqrt{\left(r_0^{2k}\left(a_0^* a_0 x_0^* x_0 + O\left(\frac{r_1}{r_0}\right)^k\right)\right)} \\
&= r_0^k \sqrt{\left(a_0^* a_0 x_0^* x_0 + O\left(\frac{r_1}{r_0}\right)^k\right)} \qquad (2.49)
\end{aligned}$$

For $\left|\Delta\bar{U}^{k-1}\right|$, similarly, we have

$$\left|\Delta\bar{U}^{k-1}\right| = r_0^{k-1} \sqrt{\left(a_0^* a_0 x_0^* x_0 + O\left(\frac{r_1}{r_0}\right)^{k-1}\right)}. \qquad (2.50)$$

The amplification factor γ is defined as the ratio of the norm of the computational errors of two successive iterations. Therefore, as k becomes large, the amplification factor at the kth iteration can be obtained:

$$\gamma_k = \frac{\left|\Delta\bar{U}^k\right|}{\left|\Delta\bar{U}^{k-1}\right|} = \frac{r_0^k \left|a_0\right| \left|x_0\right| \sqrt{\left(1 + O\left(\frac{r_1}{r_0}\right)^k\right)}}{r_0^{k-1} \left|a_0\right| \left|x_0\right| \sqrt{\left(1 + O\left(\frac{r_1}{r_0}\right)^{k-1}\right)}} = r_0 \left(1 + O\left(\frac{r_1}{r_0}\right)^{k-1}\right). \qquad (2.51)$$

As k becomes large, the term $\left(\frac{r_1}{r_0}\right)^{k-1}$ is exponentially small. Hence, we can conclude that the amplification factor $\gamma_k \to r_0$ as k becomes large. As a result, if r_0 is larger than one, the norm of $\Delta\bar{U}$ increases exponentially, resulting in instability and if is smaller than one, the norm of $\Delta\bar{U}$ decreases exponentially, resulting in stability. If the norm of r_0 is one, convergence is still not feasible. Hence, for solution

convergence, the norm of the largest eigenvalue needs to be smaller than one for the case where the largest eigenvalue is real.

We can also calculate the angle between the vector $\Delta\bar{U}^k$ and the eigenvector associated with the largest eigenvalue. The angle is defined as

$$\theta_k = \arccos\left(\frac{x_0^* \Delta\bar{U}^k}{\left|x_0^*\right| \left|\Delta\bar{U}^k\right|}\right). \tag{2.52}$$

According to Appendix B, we have

$$\theta_k = \pi \text{ or } 0 + O\left(\left(\frac{r_1}{r_0}\right)^k\right) \tag{2.53}$$

Analogous to the norm, for large k, the angle converges to 0 or π. Therefore, the two vectors converge to being parallel. Actually, this is the basis of the power method for eigenpair calculation.

(2) We also need to consider the case where the largest eigenvalues are a conjugate pair, which is common for the Jacobian matrices of flow problems, and more complicated to analyze. We can implement a similar analysis. Suppose the largest eigenvalues are a conjugate pair:

$$\lambda_0 = r_0 e^{I\theta_0},$$

$$\lambda_1 = r_0 e^{-I\theta_0}.$$

Since the computational error ΔU is real, the eigendecomposition coefficients associated with a conjugate pair should be a conjugate pair as well. Here the coefficients of eigendeomposition associated with the largest eigenvalues are presented by $a_0 e^{I\alpha_0}$ and $a_0 e^{-I\alpha_0}$. The associated eigenvectors are a conjugate pair as well. Recall Equation (2.45) and replace a_i^0 by a_0 to get

$$\Delta\bar{U}^k = A^k \Delta\bar{U} = \sum_{i=0}^{i=n-1} a_i \lambda_i^k x_i.$$

Isolating the largest eigenvalues from the rest, we get

$$\sum_{i=0}^{i=n-1} a_i \lambda_i^k x_i = r_0^k \left(a_0 e^{I(\alpha_0 + k\theta_0)} x_0 + a_0 e^{-I(\alpha_0 + k\theta_0)} \bar{x}_0 + \sum_{i=2}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right)$$

where $\bar{x}_0$ is the conjugate of x_0. We can also calculate the amplification factor of the norm of the computational error of two successive iterations as we have done for the case where the largest eigenvalue is real:

$$
\begin{aligned}
\gamma_k &= \frac{\left|\Delta\bar{U}^k\right|}{\left|\Delta\bar{U}^{k-1}\right|} = \\
&\frac{\left| r_0^k \left(\left(a_0 e^{I(\alpha_0+k\theta_0)} x_0 + a_0 e^{-I(\alpha_0+k\theta_0)} \bar{x}_0\right) + \sum_{i=2}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i \right) \right|}{\left| r_0^{k-1} \left(\left(a_0 e^{I(\alpha_0+(k-1)\theta_0)} x_0 + a_0 e^{-I(\alpha_0+(k-1)\theta_0)} \bar{x}_0\right) + \sum_{i=2}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^{k-1} e^{I(k-1)\theta_i} x_i \right) \right|} \\
&= r_0 \frac{\left| \left(a_0 e^{I(\alpha_0+k\theta_0)} x_0 + a_0 e^{-I(\alpha_0+k\theta_0)} \bar{x}_0\right) + \sum_{i=2}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i \right|}{\left| \left(a_0 e^{I(\alpha_0+(k-1)\theta_0)} x_0 + a_0 e^{-I(\alpha_0+(k-1)\theta_0)} \bar{x}_0\right) + \sum_{i=2}^{i=n-1} a_i \left(\frac{r_i}{r_0}\right)^{k-1} e^{I(k-1)\theta_i} x_i \right|}.
\end{aligned}
$$

Suppose the norm of the eigenvector equals one : $|x_0| = 1$. Therefore, we have

$$
\begin{aligned}
&\left| \left(a_0 e^{I(\alpha_0+k\theta_0)} x_0 + a_0 e^{-I(\alpha_0+k\theta_0)} \bar{x}_0\right) + \sum_{i=2}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i \right|^2 \\
&= \left(\left(a_0 e^{I(\alpha_0+k\theta_0)} x_0 + a_0 e^{-I(\alpha_0+k\theta_0)} \bar{x}_0\right) + \sum_{i=2}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i \right)^* \\
&\left(\left(a_0 e^{I(\alpha_0+k\theta_0)} x_0 + a_0 e^{-I(\alpha_0+k\theta_0)} \bar{x}_0\right) + \sum_{i=2}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i \right) \\
&= \left(\left(a_0 e^{-I(\alpha_0+k\theta_0)} \bar{x}_0^\intercal + a_0 e^{I(\alpha_0+k\theta_0)} x_0^\intercal\right) \left(a_0 e^{I(\alpha_0+k\theta_0)} x_0 + a_0 e^{-I(\alpha_0+k\theta_0)} \bar{x}_0\right) \right) + O\left(\left(\frac{r_1}{r_0}\right)^k \right) \\
&= \left(2a_0^2 + a_0^2 e^{-2I(\alpha_0+k\theta_0)} \bar{x}_0^\intercal \bar{x}_0 + a_0^2 e^{2I(\alpha_0+k\theta_0)} x_0^\intercal x_0\right) + O\left(\left(\frac{r_1}{r_0}\right)^k \right) \qquad (2.54)
\end{aligned}
$$

where the operator $()^\intercal$ is getting the transpose of a vector. In Equation (2.54), $\bar{x}_0^\intercal \bar{x}_0$ and $x_0^\intercal x_0$ are a conjugate pair of complex scalars. Given that $|x_0| = 1$, we have

$$\bar{x}_0^\intercal \bar{x}_0 = e^{-2I\beta_0},$$

$$x_0^\intercal x_0 = e^{2I\beta_0}.$$

Consequently, we have

$$\left|\left(a_0 e^{I(\alpha_0+k\theta_0)}x_0 + a_0 e^{-I(\alpha_0+k\theta_0)}\bar{x}_0\right) + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right|$$
$$= \sqrt{\left(2a_0^2 + a_0^2 e^{-2I(\alpha_0+k\theta_0+\beta_0)} + a_0^2 e^{2I(\alpha_0+k\theta_0+\beta_0)}\right) + O\left(\left(\frac{r_1}{r_0}\right)^k\right)}.$$

By applying the Euler's formula, we have

$$\sqrt{\left(2a_0^2 + a_0^2 e^{-2I(\alpha_0+k\theta_0+\beta_0)} + a_0^2 e^{2I(\alpha_0+k\theta_0+\beta_0)}\right)}$$
$$= \sqrt{\left(2a_0^2 + 2a_0^2 \cos\left(2\left(\alpha_0 + k\theta_0 + \beta_0\right)\right)\right)}.$$

thus,

$$\left|\left(a_0 e^{I(\alpha_0+k\theta_0)}x_0 + a_0 e^{-I(\alpha_0+k\theta_0)}\bar{x}_0\right) + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right|$$
$$= \sqrt{\left(2a_0^2 + 2a_0^2 \cos\left(2\left(\alpha_0 + k\theta_0 + \beta_0\right)\right)\right)} + O\left(\left(\frac{r_1}{r_0}\right)^k\right). \tag{2.55}$$

Similarly, for the iteration of $k-1$, we have

$$\left|\left(a_0 e^{I(\alpha_0+(k-1)\theta_0)}x_0 + a_0 e^{-I(\alpha_0+(k-1)\theta_0)}\bar{x}_0\right) + \sum_{i=2}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^{k-1} e^{I(k-1)\theta_i} x_i\right|$$
$$= \sqrt{\left(2a_0^2 + 2a_0^2 \cos\left(2\left(\alpha_0 + (k-1)\,\theta_0 + \beta_0\right)\right)\right)} + O\left(\left(\frac{r_1}{r_0}\right)^{k-1}\right). \tag{2.56}$$

Combing Equation (2.55) and Equation (2.56), we get

$$\begin{aligned}
\gamma_k &= \frac{\left|\Delta\bar{U}^k\right|}{\left|\Delta\bar{U}^{k-1}\right|} = r_0 \left(\frac{\sqrt{\left(2a_0^2 + 2a_0^2 \cos\left(2\left(\alpha_0 + k\theta_0 + \beta_0\right)\right)\right)} + O\left(\left(\frac{r_1}{r_0}\right)^k\right)}{\sqrt{\left(2a_0^2 + 2a_0^2 \cos\left(2\left(\alpha_0 + (k-1)\,\theta_0 + \beta_0\right)\right)\right)} + O\left(\left(\frac{r_1}{r_0}\right)^{k-1}\right)}\right) \\
&= r_0 \left(\left|\frac{\cos\left(\alpha_0 + k\theta_0 + \beta_0\right)}{\cos\left(\alpha_0 + (k-1)\,\theta_0 + \beta_0\right)}\right| + O\left(\frac{r_1}{r_0}\right)^{k-1}\right) \qquad (2.57) \\
&= r_0 \left(\left|\cos\left(\theta_0\right) - \sin\left(\theta_0\right)\tan\left(\alpha_0 + (k-1)\,\theta_0 + \beta_0\right)\right| + O\left(\left(\frac{r_1}{r_0}\right)^{k-1}\right)\right). \qquad (2.58)
\end{aligned}$$

Dropping the high order terms in equation (2.58), we have

$$\gamma_k = \gamma(k, \theta_{0,} \alpha_{0,} \beta_0) = r_0 \left|\cos(\theta_0) - \sin(\theta_0) \tan(\alpha_0 + (k-1)\theta_0 + \beta_0)\right|. \quad (2.59)$$

Unlike the case where the largest eigenvalue is real, in which the amplification factor of the norm of $\Delta\bar{U}$ of two successive iterations converges to the norm of the largest eigenvalue, in this case, the amplification factor associated with two successive iterations also depend on the number of iterations k, and from Equation (2.59), we can see the amplification factor will oscillate around r_0 since $\tan()$ is a periodic function. Since stability describes the asymptotic properties of a scheme, a small number of iterations is not informative. On the contrary, we must focus on the asymptotic ratios of the norm of the computational error over a large number of iterations. With the help of Equation (2.57), we can calculate the amplification factor of the norm of the computational error of two largely distant iterations. Assume that if $k > m$, the higher order terms in Equation (2.57) can be ignored. We define

$$\Gamma_k = \frac{\left|\Delta\bar{U}^k\right|}{\left|\Delta\bar{U}^0\right|}. \quad (2.60)$$

For a number $n > m$, we have

$$\Gamma_n = \frac{\left|\Delta\bar{U}^n\right|}{\left|\Delta\bar{U}^0\right|} = \Gamma_m \prod_{k=m+1}^{k=n} \gamma_k. \quad (2.61)$$

Recalling Equation (2.57) for γ and dropping the high order terms, we have

$$\Gamma_n = \Gamma_m \prod_{k=m+1}^{k=n} \left|\frac{\cos(\alpha_0 + k\theta_0 + \beta_0)}{\cos(\alpha_0 + (k-1)\theta_0 + \beta_0)}\right| = \Gamma_m r_0^{n-m} \left|\frac{\cos(\alpha_0 + n\theta_0 + \beta_0)}{\cos(\alpha_0 + m\theta_0 + \beta_0)}\right|. \quad (2.62)$$

From Equation (2.62), we can easily conclude that r_0 determines the asymptotic rate of the growth of the computational error $\Delta\bar{U}$. Therefore, to have a stable system, the following condition is needed:

$$r_0 < 1. \quad (2.63)$$

If r_0 is smaller than one, the scheme is stable. However, from Equation (2.59), we expect that the norm of the computational error $\Delta\bar{U}$ will increase locally, which should explain the phenomena mentioned in F. Haider et al (2009) [7] that the energy might increase locally even if the scheme is stable. For the direction of

the computational error $\Delta\bar{U}$, it is complicated enough that we do not discuss it here. Though here we only analyze the computational error, for the other two computational vectors: the flux integral and the solution update, the conclusions should be the same.

2.3 Stability of the Method of Lines

In Section 2.2, we presented the stability condition for the full discretization and relates the behavior of the computational vectors to the eigenvalues of the mapping. In practice, it is more usual to focus on the stability properties of the spatial discretization, i.e., the method of lines. In this section, we present the stability condition for the spatial discretization.

The semi-discretization of the 2D Euler equations can be also written as

$$\frac{d\bar{U}(t)}{dt} = -R(\bar{U}(t)). \tag{2.64}$$

Since this equation is nonlinear, we focus on the stability of the steady state solution, i.e., $\bar{U}_s$, which is also the fixed point. For a small perturbation to the steady solution $\bar{U}_p = \bar{U}_s + \Delta\bar{U}_p$, we have

$$\frac{d\Delta\bar{U}_p(t)}{dt} = -\frac{\partial R}{\partial \bar{U}_s}\Delta\bar{U}_p(t) + h(\Delta\bar{U}_p(t),\, t)$$

where $h(\Delta\bar{U}_p(t),\, t)$ is a function representing the difference between $\frac{d\Delta\bar{U}_p(t)}{dt}$ and $-\frac{\partial R}{\partial \bar{U}_p}\Delta\bar{U}_p(t)$. Theorem 4.3 in [1] is on the stability of a system like Equation (2.64). We follow this theorem to discuss the stability of the fixed point solution of Equation (2.64). We assume $h(\Delta\bar{U}_p(t),\, t)$ and $-\frac{\partial R}{\partial \bar{U}_s}$ are continuous in $(t,\, \Delta\bar{U}_p)$ for $0 \le t < \infty$, $|\Delta\bar{U}_p| < k$ where k is a small positive constant. We also assume for $h(\Delta\bar{U}_p(t),\, t)$, the following condition is satisfied:

$$\lim_{|\Delta\bar{U}_p|\to 0} \frac{|h(\Delta\bar{U}_p(t),\, t)|}{|\Delta\bar{U}_p(t)|} = 0, \qquad \text{uniformly for t} \in [0,\infty).$$

If we need to have a stable fixed point solution, we need all the eigenvalues of matrix $-\frac{\partial R}{\partial \bar{U}_s}$ to be negative, i.e., $Re\left(\Lambda\left(-\frac{\partial R}{\partial \bar{U}_s}\right)\right) < 0$, where $\Lambda\left(-\frac{\partial R}{\partial \bar{U}_s}\right)$ means the spectrum of the Jacobian matrix $-\frac{\partial R}{\partial \bar{U}_s}$.

We need to use the stability condition of semi-discretization for stabilization at the fixed point and a non-fixed point. For both cases, $h(\Delta\bar{U}_p(t),\, t)$ is small; therefore, it is sufficient to use $Re\left(\Lambda\left(-\frac{\partial R}{\partial \bar{U}_s}\right)\right) < 0$ as stability condition.

Chapter 3

Numerical Validation for Stability Methodology

3.1 Numerical Testing Methodology

In this chapter, we test the validity of the stability analysis methodology of Chapter 2 by numerical examples. We use two linear problems: Poisson's equation and the advection-diffusion equation; and a nonlinear problem: the two dimensional Euler equations as test problems. For Poisson's equation and the advection diffusion equation, least squares solution reconstruction [19] is employed. For the two dimensional Euler equations, least squares reconstruction [19] and Roe flux difference splitting [24] are used. The spatial discretization of the PDEs is implemented by using the Advanced Numerical Simulation Library (ANSLib) [21]. We use the method in [17] to implement pseudo-timestepping to reach the steady state solution. The implementation of the time-stepping employs the Portable, Extensible Toolkit for Scientific Computation [25] (PETSc), a linear algebra library, to reach the converged solution. Once the converged solution is reached, we perturb the solution randomly, and apply backward Euler time-stepping. For mesh generation, we use the Generation and Refinement of Unstructured, Mixed-Element Meshes in Parallel (GRUMMP) [20] package.

We also need the explicit formation of the Jacobian matrix and its eigenvalues to test the methodology. The Jacobian matrix is calculated analytically by the method originally developed by Michalak and Olliver-Gooch (2010) [17] and stored explicitly. The eigenvalue calculation is implemented by the Scalable Library for Eigenvalue Problem Computations [2] (SLEPc).

The mappings of the solution update, the computational error, and the flux integral are identical if the backward Euler time stepping method is employed. As a result, the stability conditions inherited from these three mappings are the same, which is reasonable. In Chapter 2, we have developed a model for the asymptotic behavior of three variables: the solution update $\delta\bar{U}$, the computational error $\Delta\bar{U}$, and the flux integral $R(\bar{U})$, and also constructed the corresponding stability condition; we have also established a stability condition based solely on the spatial

discretization. In this chapter, we test if the prediction of the behavior of those variables based on the presented model is consistent with their actual behavior. If the prediction coincides with the actual behavior, the methodology is valid.

The three variables, solution update, computational error, and flux integral, are all real vectors. As a vector, the norm and the direction are of interest to us. Mapping (2.33) models how the norm and direction change after a large number of iterations, i.e., asymptotically. In previous analysis, we have drawn the following conclusions. (1) If the largest eigenvalue is real, as the iteration becomes a large number, the amplification factor converges to the norm of the largest eigenvalue for each variable, and the direction of the three vectors converge to being parallel to the direction of the eigenvector associated with the largest eigenvalue. (2) If the largest eigenvalues are a conjugate complex pair, the norm and the direction behave differently from the case where the largest eigenvalue is real. For this case, neither the norm nor the direction will go to a constant as the iteration k becomes a large number. We use a different approach to test the validity indirectly.

Recall Equation (2.62), and take the high order term into consideration:

$$\Gamma_n = \frac{|\Delta\bar{U}^n|}{|\Delta\bar{U}^0|} = \Gamma_m \left(r_0^{n-m} \left| \frac{\cos(\alpha_0 + n\theta_0 + \beta_0)}{\cos(\alpha_0 + m\theta_0 + \beta_0)} \right| + O\left(\frac{r_1}{r_0}\right)^m \right). \tag{3.1}$$

Thus, for two large numbers: j, k, the ratio of the norm of the computational error of these two iterations is

$$\Gamma_{k-j} = \frac{\Delta\bar{U}^k}{\Delta\bar{U}^j} = r_0^{k-j} \left| \frac{\cos(\alpha_0 + k\theta_0 + \beta_0)}{\cos(\alpha_0 + j\theta_0 + \beta_0)} \right| + O\left(\frac{r_1}{r_0}\right)^j. \tag{3.2}$$

Suppose $(k-j)\,\theta_0 = l\pi + \phi$, where l is a integral and $|\phi| < |\theta_0| < \pi$. Thus, we have

$$\Gamma_{k-j} = \frac{\Delta\bar{U}^k}{\Delta\bar{U}^j} = r_0^{k-j} \left| \frac{\cos(\alpha_0 + j\theta_0 + \beta_0 + l\pi + \phi)}{\cos(\alpha_0 + j\theta_0 + \beta_0)} \right| + O\left(\frac{r_1}{r_0}\right)^j. \tag{3.3}$$

in which, we have

$$\begin{aligned}
\left| \frac{\cos(\alpha_0 + j\theta_0 + \beta_0 + l\pi + \phi)}{\cos(\alpha_0 + j\theta_0 + \beta_0)} \right| &= \left| \frac{\cos(\alpha_0 + j\theta_0 + \beta_0 + \phi)}{\cos(\alpha_0 + j\theta_0 + \beta_0)} \right| \\
&= \left| \frac{\cos(\alpha_0 + j\theta_0 + \beta_0)\cos(\phi) - \sin(\alpha_0 + j\theta_0 + \beta_0)\sin(\phi)}{\cos(\alpha_0 + j\theta_0 + \beta_0)} \right| \\
&= |\cos(\phi) - \tan(\alpha_0 + j\theta_0 + \beta_0)\sin(\phi)| \\
&\approx 1 - \frac{1}{2}\phi^2 - \tan(\alpha_0 + j\theta_0 + \beta_0)\left(\phi - \frac{1}{6}\phi^3\right).
\end{aligned} \tag{3.4}$$

Therefore, we have

$$\Gamma_{k-j} = \frac{\Delta\bar{U}^k}{\Delta\bar{U}^j} \approx r_0^{k-j}\left(1 - \frac{1}{2}\phi^2 - \tan\left(\alpha_0 + j\theta_0 + \beta_0\right)\phi\right) + O\left(\frac{r_1}{r_0}\right)^j. \tag{3.5}$$

We define an averaged amplification factor of two successive iterations as

$$\bar{\gamma} = \Gamma_{k-j}^{\frac{1}{k-j}}. \tag{3.6}$$

The terms, $\frac{1}{2}\phi^2$, $\tan\left(\alpha_0 + j\theta_0 + \beta_0\right)\phi$, and $O\left(\frac{r_1}{r_0}\right)^j$ are small. Substitute Equation (3.5) into Equation (3.6) to give

$$\bar{\gamma} = r_0\left(1 + O\left(\frac{-\frac{1}{2}\phi^2 - \tan\left(\alpha_0 + j\theta_0 + \beta_0\right)\phi}{k-j}\right)\right) + O\left(\frac{1}{k-j}\left(\frac{r_1}{r_0}\right)^j\right). \tag{3.7}$$

Given Equation (3.7), we can therefore test if $\bar{\gamma}$ matches r_0 to test the validity of the methodology. From Equation (2.59), we can also see that the convergence history will oscillate, and the amplification factor associated with two successive iterations will oscillate around the norm of the largest eigenvalues. As it is complicated to analyze the variation of the direction of the computational error, solution update, and flux integral for large iteration count, we do not implement numerical tests for it.

Since the direction of the three vectors is evaluated by the angle between the vector and the eigenvector associated with the largest eigenvalue, we use the term angle to replace the term direction.

3.2 Precision Estimation

It would be misleading to state that one set of data matches another set of data, while the degree of coincidence is not explicitly presented. In this section, we estimate analytically the extent to which the actual data matches the analytical value both for the amplification factor and the angle. We begin with estimating the precision of the model.

For a linear problem, for instance, Poisson's equation, the derivation of Mapping (2.31) does not need to drop the higher order terms; thus, the mapping precisely describes the relation of the three variables — computational error, solution update, and flux integral — between two successive iterations. For a nonlinear problem, specifically the Euler equations in this thesis, the dropped higher order

terms are of $O\left(\left(\Delta\bar{U}^k\right)^2\right)$, which is sufficiently small that it is not the principal factor that affects the precision.

Next, we need to estimate how close the actual amplification factor will be with the expected value — the norm of the largest eigenvalue, and estimate how close the angle will be with the expected value (0 or π) as well. Again, here we divide the discussion into two categories: one is the case where the largest eigenvalue is real, and the other is the case where the largest eigenvalues are a conjugate complex pair.

(1) We consider first the case where the largest eigenvalue is real. Recalling Equation (2.51), we have

$$\gamma_k = r_0\left(1 + O\left(\frac{r_1}{r_0}\right)^{k-1}\right). \tag{3.8}$$

Seen from this equation, the extent to which the actual amplification factor coincides with r_0 is constrained primarily by the term $O\left(\frac{r_1}{r_0}\right)^{k-1}$. We use $\Delta\gamma_k$ to denote the difference between the actual amplification factor at the kth iteration γ_k and r_0:

$$\Delta\gamma_k = |\gamma_k - r_0|. \tag{3.9}$$

$\Delta\gamma$ refers to the difference between the actual amplification factor and r_0 when the iteration is not specifically expressed. From Equation (3.8), we can see that $\Delta\gamma_k$ behaves similarly as $\left(\frac{r_1}{r_0}\right)^{k-1}$. Simply put if $\Delta\gamma_k$ is small, which means γ_k is close to r_0, and the decay rate of $\Delta\gamma_k$ is close to $\frac{r_1}{r_0}$, Equation (3.8) holds, which verifies the validity of the stability analysis model.

For the angle, we have similar conclusions. Recall Equation (2.53)

$$\theta_k = \pi \text{ or } 0 + O\left(\left(\frac{r_1}{r_0}\right)^k\right).$$

The difference between the actual angle and π or 0 is denoted by $\Delta\theta$, which for the kth iteration is

$$\Delta\theta_k = |\theta_k| \text{ or } |\theta_k - \pi| = O\left(\left(\frac{r_1}{r_0}\right)^k\right). \tag{3.10}$$

Similar to $\Delta\gamma_k$, if $\Delta\theta_k$ is small and the decay rate is close to $\frac{r_1}{r_0}$, this verifies the validity of Equation (2.53), and therefore the validity of the stability analysis model.

(2) We consider the case where the largest eigenvalues are complex. For this case, we only study the amplification factor. Retrieve Equation (3.7):

$$\bar{\gamma} = r_0 \left(1 + O\left(\frac{-\frac{1}{2}\phi^2 - \tan\left(\alpha_0 + j\theta_0 + \beta_0\right)\phi}{k-j} \right) \right) + O\left(\frac{1}{k-j} \left(\frac{r_1}{r_0} \right)^j \right) \quad (3.11)$$

From this equation, We see that the precision in which $\bar{\gamma}$ matches r_0 is constrained by the larger term between $O\left(\frac{-\frac{1}{2}\phi^2 - \tan(\alpha_0 + j\theta_0 + \beta_0)\phi}{k-j}\right)$ and $O\left(\frac{1}{k-j}\left(\frac{r_1}{r_0}\right)^j\right)$. Both terms are complicated to evaluate, but if the difference between $\bar{\gamma}$ and r_0 is small, we can be confident of the validity of the model for the case where the largest eigenvalues are complex.

3.3 Test Case No. 1 — Poisson's Equation

The form of Poisson's equation is

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = S, \quad (3.12)$$

where

$$S = 2\pi^2 \sin(\pi x) \sin(\pi y). \quad (3.13)$$

The spatial domain is a unit square. The boundary condition is $u = 0$ at the boundary.

Due to the inherent linearity, Equation (2.32) holds exactly for Poisson's equation since no higher order terms are dropped. The problem is solved on an unstructured mesh consisting of 784 triangles. Cell-centered control volumes are employed for flux integration. The fourth order reconstruction scheme is used.

The mesh on that the equation is solved is shown in Figure (3.1), and the solution is shown in Figure (3.2). Once the converged solution is reached, we perturb the solution randomly, and then we apply backward Euler time-stepping with $dt = 0.01$, starting from the perturbed solution. The first twenty largest eigenvalues are shown in Figure (3.3), in which the unit circle is also shown. If all the eigenvalues lie inside the unit circle, the scheme is stable; otherwise it is not stable.

3.3.1 Numerical Results and Data Analysis

As demonstrated previously, the mappings of the three variables are identical as long as the backward Euler temporal discretization with a constant time-step is

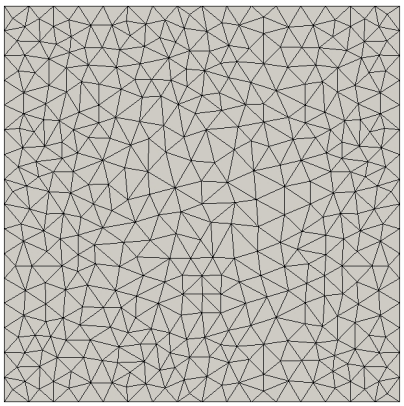


Figure 3.1: Mesh for solving Poisson's equation

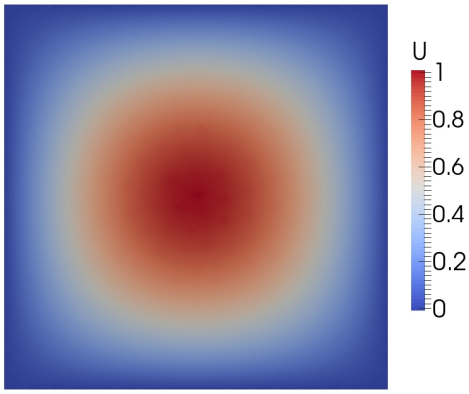


Figure 3.2: Solution to Poisson's equation

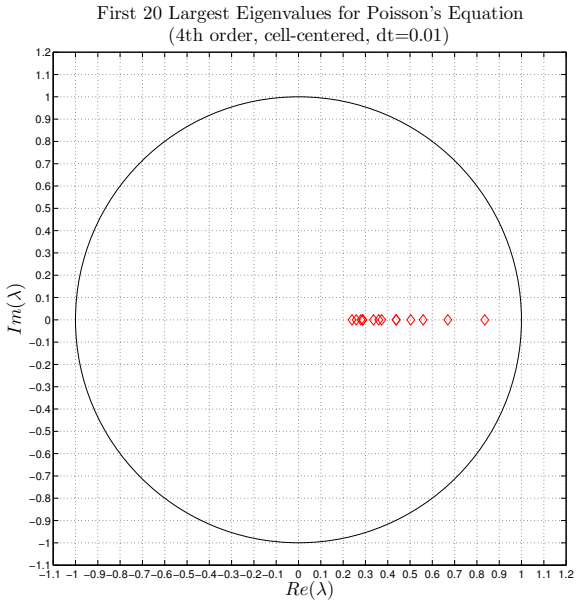


Figure 3.3: First twenty largest eigenvalues for Poisson's equation case

applied, and the convergence rate, i.e., the amplification factor, is supposed to be the norm of the largest eigenvalue. Therefore, we can plot the convergence histories for all three variables, and the convergence rate predicted by the eigenvalue as well for comparison. To obtain a sophisticated sense of the convergence rates or amplification factors, we plot the amplification factors of two successive iterations as well.

As seen in Figure (3.4), the convergence histories of the flux integral, the computational error, and the solution update are consistent with the convergence rate predicted by the largest eigenvalue. We also compare the amplification factors associated with two successive iterations with the norm of the largest eigenvalue in Figure (3.5). From Figure (3.5), we can see that after the 25th iteration, the amplification factors agree well with the norm of the largest eigenvalue for all three variables. For the flux integral, we can see that the degree of coincidence decreases after the 40th iteration, as the computational error approaches the machine precision. Given Equation (A.10) in Appendix A, it is reasonable that the amplification factor of the flux integral coincides with the amplification factor of the solution update.

To have more clear understanding of how close the amplification factors are to the norm of the largest eigenvalue, we calculate $\Delta\gamma_k$, and plot this to obtain Figure (3.6). From Figure (3.6), we can draw the following conclusions: (1) for all three variables, the norm of $\Delta\gamma_k$ is small after the 25th iteration. After the 40th iteration, for the flux integral, the norm of $\Delta\gamma_k$ increases. The decay rates of $\Delta\gamma_k$ are consistent with $\frac{r_1}{r_0}$. However, for the flux integral, the growth rate after the 40th iteration does not coincide well with $\frac{r_1}{r_0}$.

Similar to the amplification factor, for the angle, we calculate $\Delta\theta_k$ to obtain Figure (3.7) . We can see that for all three variables, $\Delta\theta_k$ is small. For the computational error, the decay rate of $\Delta\theta_k$ is consistent with $\frac{r_1}{r_0}$ from the 25th to the 44th iteration. For the solution update, the decay rate of $\Delta\theta_k$ is consistent with $\frac{r_1}{r_0}$ from 25th iteration to the 42th iteration. For the flux integral, the decay rate of $\Delta\theta_k$ is consistent with $\frac{r_1}{r_0}$ from the 25th iteration to the 33th iteration. The precision of coincidence decreases as the computational error approaches the machine precision.

We also visualize the eigenvector associated with the largest eigenvalue in Figure (3.8a). The vector of the flux integral at the 30th iteration is plotted in Figure (3.8b). Figure (3.8c) is the image of the solution update at the 45th iteration, and Figure (3.8d) is the image of the vector of the computational error at the 45th iteration. We can observe that their shapes are similar and the direction of the solution update is opposite from the other three variables, which is expected from Equation (A.10).

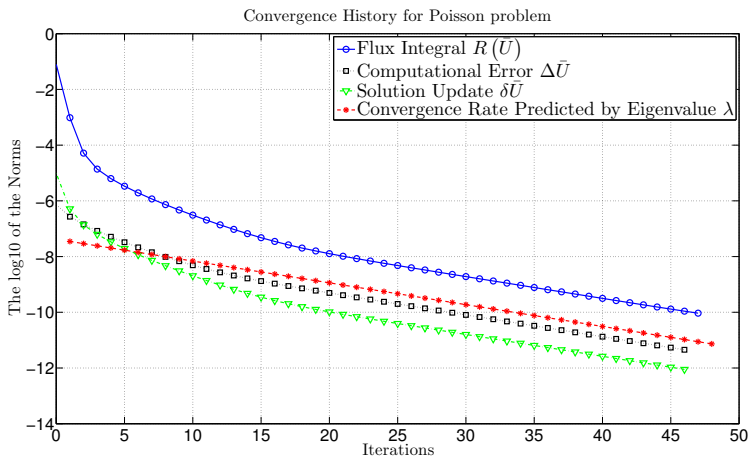


Figure 3.4: Convergence history for Poisson's equation

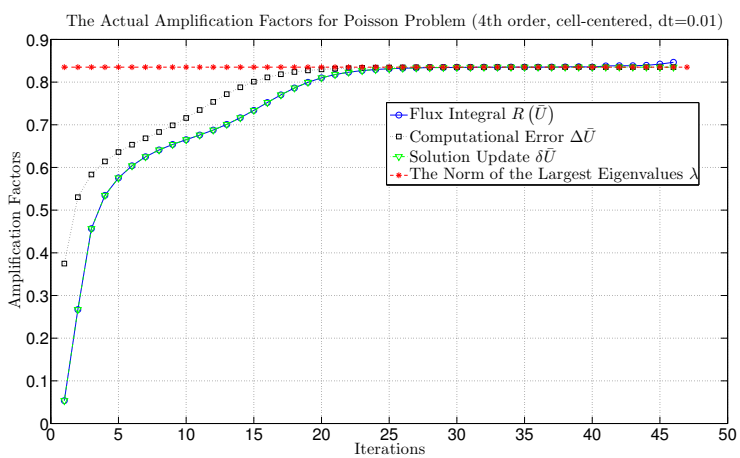


Figure 3.5: Amplification factors for Poisson's equation

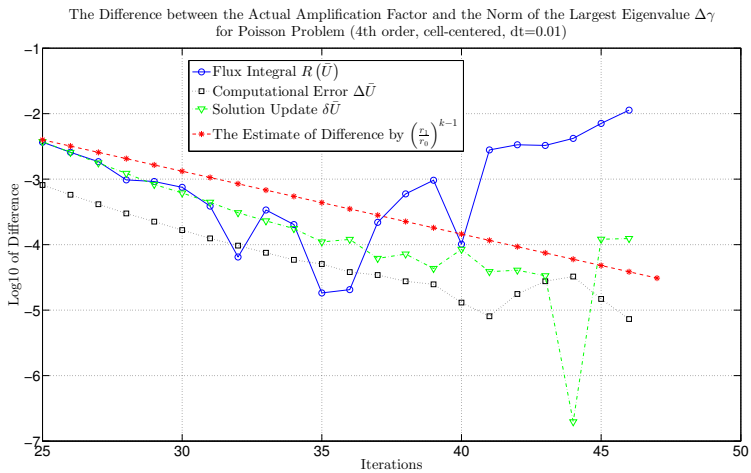


Figure 3.6: The difference between the actual amplification factor and the norm of the largest eigenvalue ($\Delta\gamma$) for Poisson problem

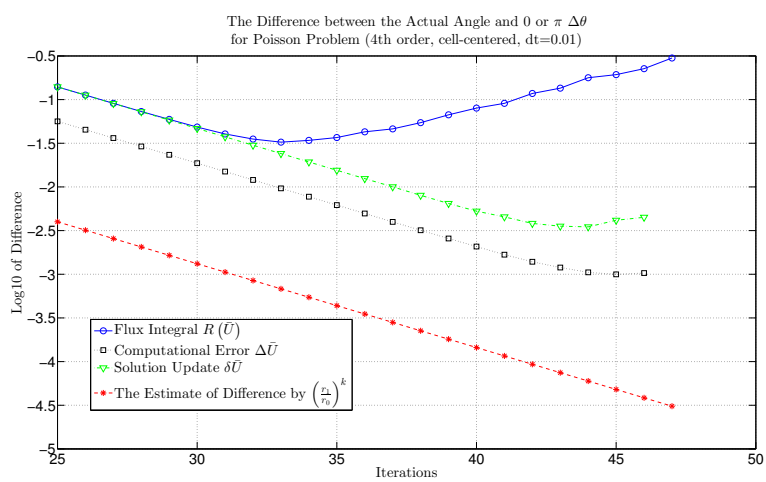
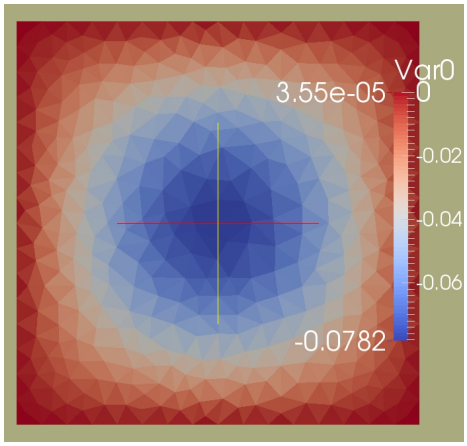
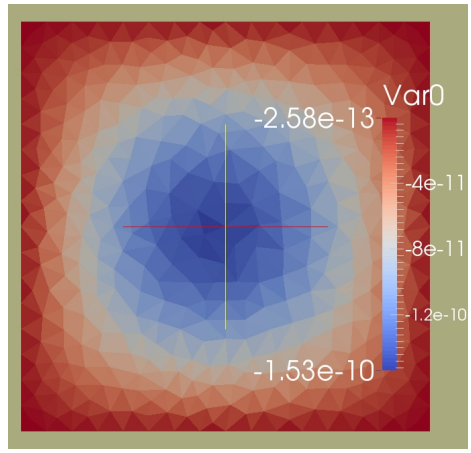


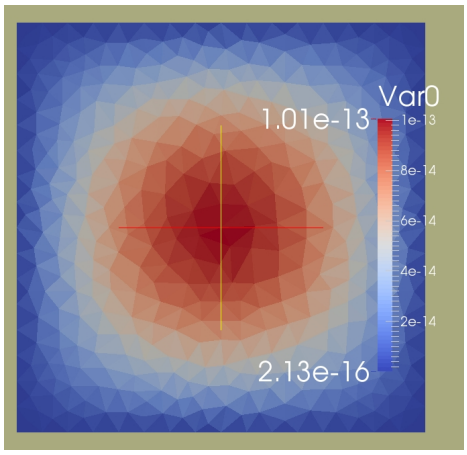
Figure 3.7: The difference between the actual angle and 0 or π ($\Delta\theta$) for Poisson problem



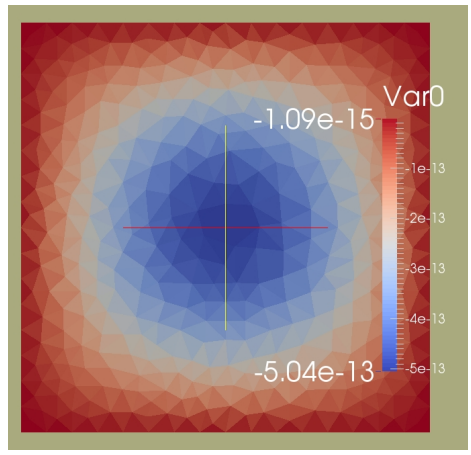
(a) The eigenvector associated with the largest eigenvalue



(b) The flux integral at the 30th iteration



(c) The solution update at the 45th iteration



(d) The computational error at the 45th iteration

Figure 3.8: Comparison among computational vectors and eigenvector for Poisson's equation case

3.3.2 Summary

The data and analysis above give us the following conclusions:

1. For the amplification factor: $\Delta\gamma_k$ is small for all three variables and the decay rate of $\Delta\gamma_k$ is consistent with $\frac{r_1}{r_0}$ for all three variables.
2. For the angle: $\Delta\theta_k$ is small for all three variables and the decay rate of $\Delta\gamma_k$ is consistent with $\frac{r_1}{r_0}$ for all three variables.

Combined with the numerical validation methodology discussed previously, these conclusions demonstrate the validity of the stability methodology strongly for Poisson's equation. In this case, the largest eigenvalue is real. In the next section, we consider the case where the largest eigenvalues are complex.

3.4 Test Case No. 2 — 2D Advection-Diffusion Problem

Compared with Poison's equation, the Jacobian matrix associated with this case features a conjugate complex pair of largest eigenvalues, which is also often seen in the Jacobian matrix associated with a flow problem. The equation for the 2D advevtion-diffusion problem is

$$\frac{\partial u}{\partial t} + a(x, y)\frac{\partial u}{\partial x} + b(x, y)\frac{\partial u}{\partial y} = \alpha\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right), \tag{3.14}$$

where $\alpha = 0.005$, $a(x, y) = 1.0$, $b(x, y) = 0.0$. The spatial domain is a rectangle with the length L equal to three in the x-direction and with the height H equal to one in the y-direction. The boundary conditions are

$$u(0, y) = \sin(\pi y),$$

$$u(x, 0) = u(x, H) = 0,$$

$$\frac{\partial u(L, y)}{\partial x} = 0.$$

Backward Euler time-stepping with $dt = 0.01$, fourth order reconstruction, and cell-centered control volumes are employed. The mesh, which is shown in Figure (3.9), contains 222 triangles. The solution is presented in Figure (3.10). The first twenty largest eigenvalues are shown in Figure (3.11).

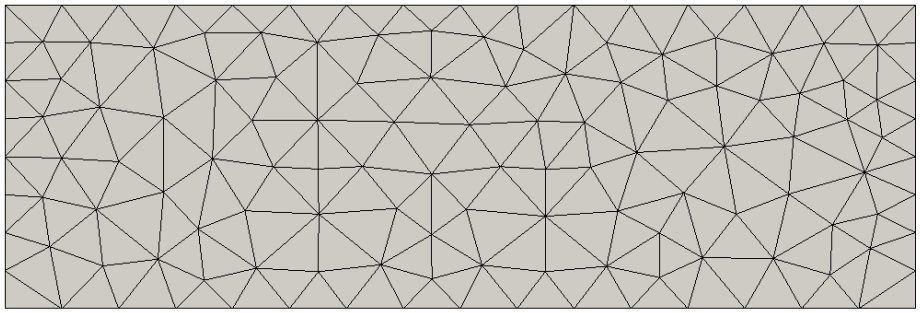


Figure 3.9: Mesh for advection-diffusion problem

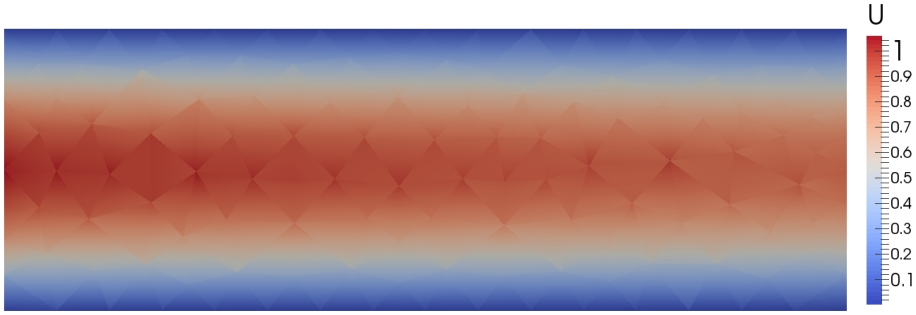


Figure 3.10: Solution to advection-diffusion problem

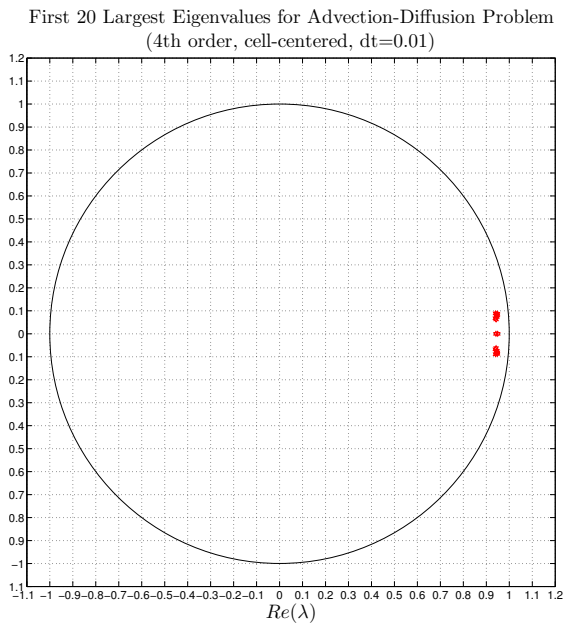


Figure 3.11: First twenty largest eigenvalues for the advection-diffusion problem

3.4.1 Numerical Results and Data Analysis

As in the last case, we generate the image of convergence histories in Figure (3.12), and the visualization of amplification factors in Figure (3.13). Different from the last case, in this case, we see the convergence history oscillate, which is predicted in theory, but the asymptotic rates of the convergence are consistent with the rate predicted by the norm of the largest eigenvalue. From Figure (3.13), we see the amplification factors oscillate around the norm of the largest eigenvalue, which is consistent with the previous discussion. Figure (3.14) is a zoom of of the part of amplification factor from the 580th iteration to the 697th iteration. From Figure (3.14), we see the amplification factors behave roughly periodically.

According to the previous discussion, we can calculate the period in the following way. First, the angle associated with the largest eigenvalues is

$$\theta = \arctan\left(\frac{Im\,(\lambda_0)}{Re\,(\lambda_0)}\right) = 0.09193. \tag{3.15}$$

The approximate period for amplification factor is

$$T = \frac{\pi}{\theta} \approx 34 \tag{3.16}$$

This period agrees with period measured in Figure (3.14).

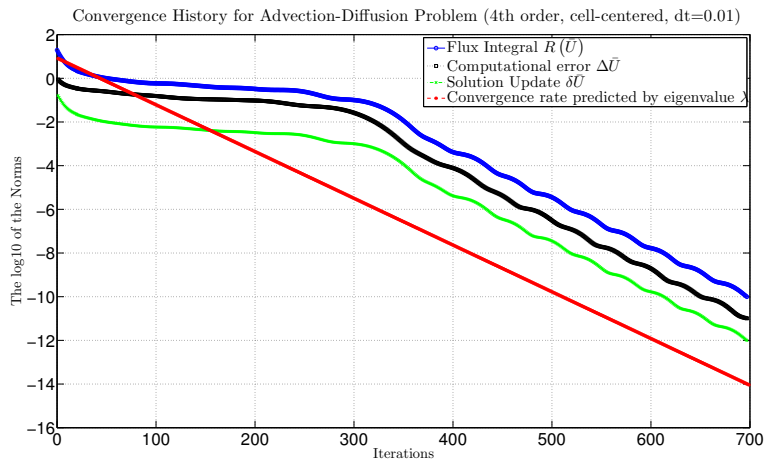


Figure 3.12: Convergence history for the advection-diffusion problem

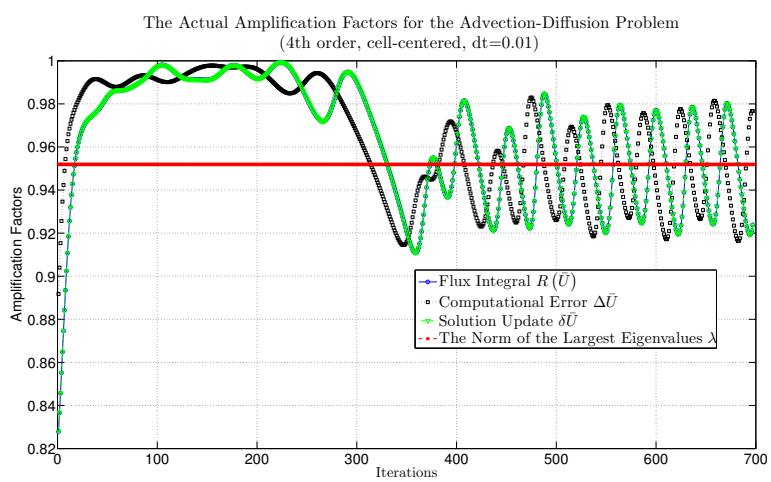


Figure 3.13: Amplification factors of two successive iterations for the advection-diffusion problem

To investigate the asymptotic behavior of the amplification factors, we calculate the average amplification factors defined by Equation (3.6). The procedure is:

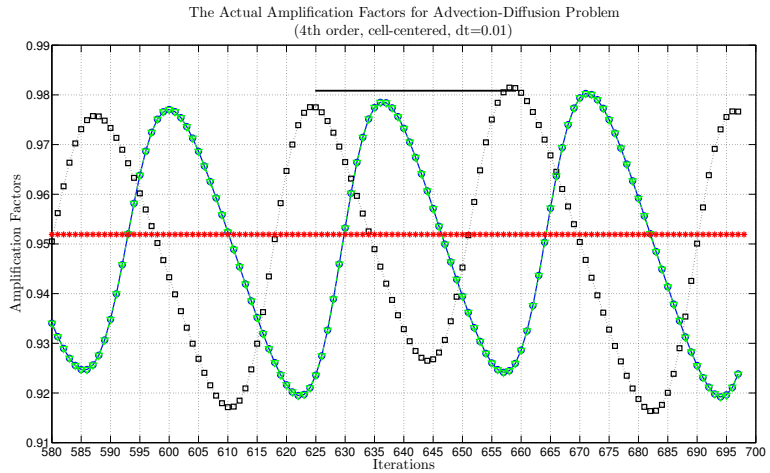


Figure 3.14: A closeup of Figure (3.13)

1. Calculate the total amplification factors for two iterations k, $k+T$:

$$\Gamma_{total} = \frac{\left|\Delta\bar{U}^{k+T}\right|}{\left|\Delta\bar{U}^{k}\right|}. \tag{3.17}$$

2. Calculate the average amplification factor

$$\bar{\gamma} = \Gamma_{total}^{\frac{1}{T}}.$$

3. Repeat (1) - (2) for a range of k to obtain a mapping of average amplification factors with respect to the iteration count k, which is visualized in Figure (3.15), and please note that the scale is different from Figure (3.13) and Figure (3.14)

In Figure (3.15), k starts from 500, and $T = 34$. Although the averaged amplification factors still oscillate, the difference is much more smaller and the largest difference is below 0.01. Therefore, the validity of Equation (3.11) is verified.

3.4.2 Summary

Because the largest eigenvalues are complex, we only consider the amplification factors. From the data and analysis in this section, we can draw the following conclusions:

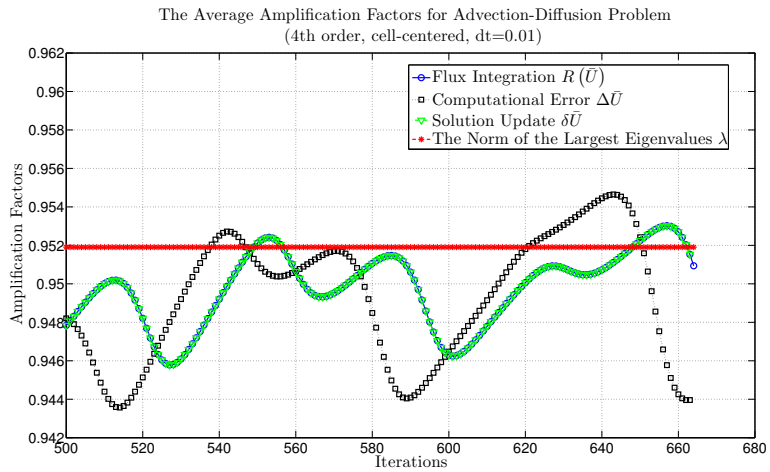


Figure 3.15: The average amplification factors of 34 iterations for the advection diffusion problem

1. As predicted in theory, the convergence rate of the three variables does not converge to a constant, but the asymptotic rate is consistent with the norm of the largest eigenvalue.

2. The amplification factors of two successive iterations oscillate around the norm of the largest eigenvalue, which coincides with the theoretic prediction.

3. The amplification factors of two successive iterations behave periodically with the predicted period.

4. The average amplification factor defined in this chapter is close to the norm of the largest eigenvalue.

For the case where the largest eigenvalues are complex, we have Equation (2.57), Equation (2.58), and Equation (2.59) to calculate the amplification factors of two successive iterations. Though we do not have direct evidence of the validity of these three equations, the conclusions (1) – (4) can show the validity of these three equations indirectly.

3.5 Test Case No. 3 — One Stable 2D Euler Equations

The previous two cases demonstrate the validity of the methodology for linear problems, for both real and complex largest eigenvalues. However, the problem of

interest is the two dimensional Euler equations. In the following sections, we test the validity of the methodology applied on the discretized time-stepping systems inherited from the two dimensional Euler equations. We begin with a stable case where the largest eigenvalue is real.

The governing equations are the 2D Euler equations, which are shown in Equation (2.1), Equation (2.2), and Equation (2.3). The spatial domain is the NACA 0012 airfoil. Slip boundary condition is applied along the the airfoil; the free stream is the steady flow with Mach number equal to 0.75; and the attack angle for the airfoil is 0.5 degrees.

The fourth order reconstruction, Roe flux difference splitting, cell-centered control volumes, and the backward Euler time stepping with $dt = 10$ are used. The mesh, shown in Figure (3.16), contains 3063 triangles and the zoom of the part around the airfoil is shown in Figure (3.17). The solution in terms of Mach field is presented in Figure (3.18), and the zoom of the Mach field around the airfoil is shown in Figure (3.19). The first twenty largest eigenvalues are plotted in Figure (3.20).

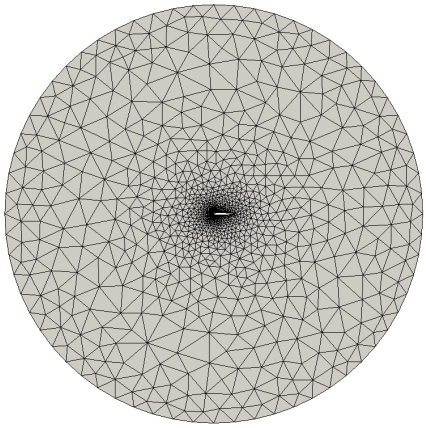


Figure 3.16: Full image of the mesh for solving the two dimensional Euler case with Mach=0.75, angle=0.5

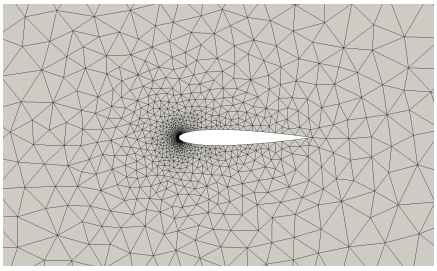


Figure 3.17: Zoom of the mesh around the airfoil for solving the two dimensional Euler case with Mach=0.75, angle=0.5

3.5.1 Numerical Results and Data Analysis

Figure (3.21) shows the convergence histories. The amplification factors associated with two successive iterations are shown in Figure (3.22). Both figures show that the actual convergence rates coincide well with the norm of the largest eigenvalue.

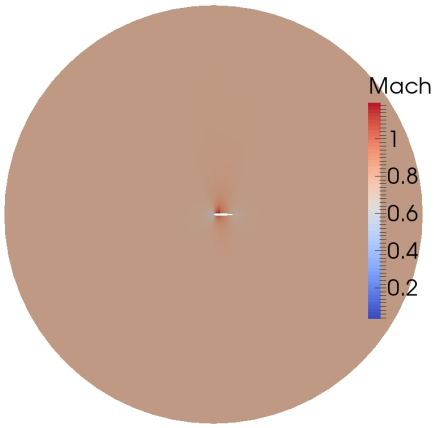


Figure 3.18: Mach field for the two dimensional Euler equations with Mach=0.75, angle=0.5 free stream

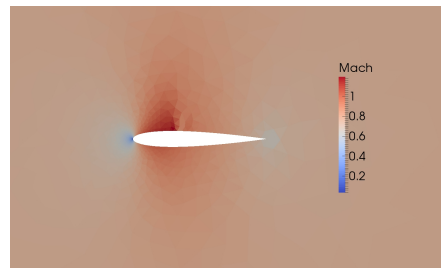


Figure 3.19: Zoom of the Mach field around the airfoil for the two dimensional Euler equations with Mach=0.75, angle=0.5 free stream

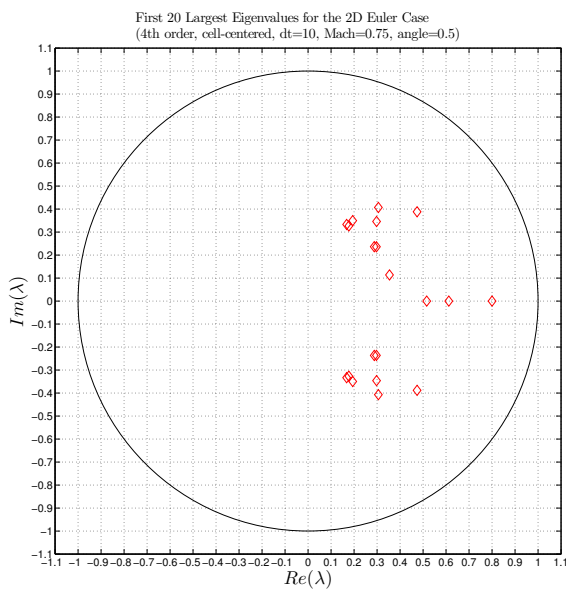


Figure 3.20: The first twenty largest eigenvalues for the two dimensional Euler case with Mach=0.75, angle=0.5

The difference between the actual amplification factors and the norm of the largest eigenvalue $\Delta\gamma$ with respect to the iteration count is plotted in Figure (3.23). From Figure (3.23), We see that the $\Delta\gamma$ is small for all three variables and roughly it is of $O\left(\frac{r_1}{r_0}\right)^{k-1}$. We also see the oscillation of $\Delta\gamma$ in Figure (3.23). The second largest eigenvalues are a conjugate pair, which explains the oscillation of $\Delta\gamma$. However, the asymptotic decay rates of $\Delta\gamma$ coincide with $\frac{r_1}{r_0}$. For the flux integral, we see the precision decreases when the computational error approaches machine precision.

For the angle, we calculate $\Delta\theta$ to obtain Figure (3.24). We can see $\Delta\theta$ is small after the 31th iteration. A pattern of oscillation can be seen in $\Delta\theta$ for all three variables but the asymptotic rates are consistent with $\frac{r_1}{r_0}$. Here we also see the precision of coincidence decreases near convergence.

We also present the visualization of the vectors (Figure (3.25a), Figure (3.25b), Figure (3.25c), and Figure (3.25d)). There are four variables in each control volume, but only the density vector is displayed. Since the entries with relatively large norm are all near the airfoil, only the zoom of the area near the airfoil is presented. We can see that the shapes are similar and the solution update vector is opposite in direction to other three vectors.

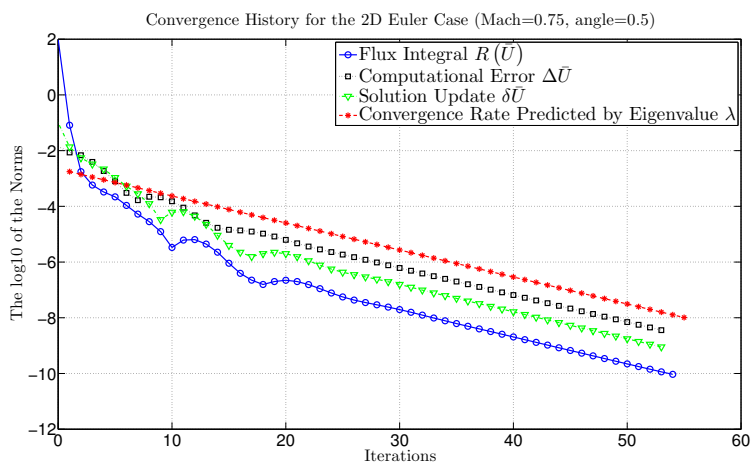


Figure 3.21: Convergence history for the two dimensional Euler case with Mach=0.75, angle=0.5

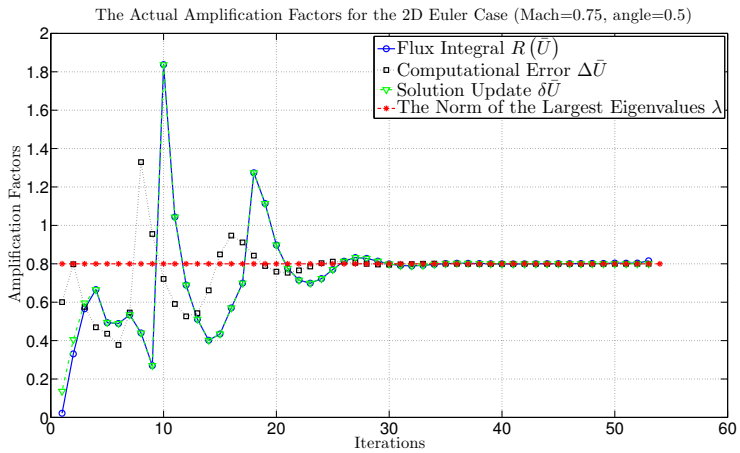


Figure 3.22: Amplification factor comparison for the two dimensional Euler case with Mach=0.75, angle=0.0

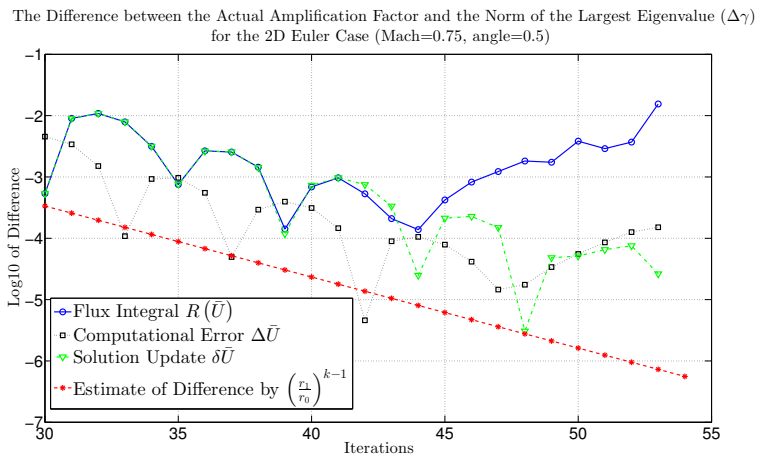


Figure 3.23: The difference between the amplification factor and the norm of the largest eigenvalue ($\Delta\gamma$) for the two dimensional Euler case with Mach=0.75, angle=0.0

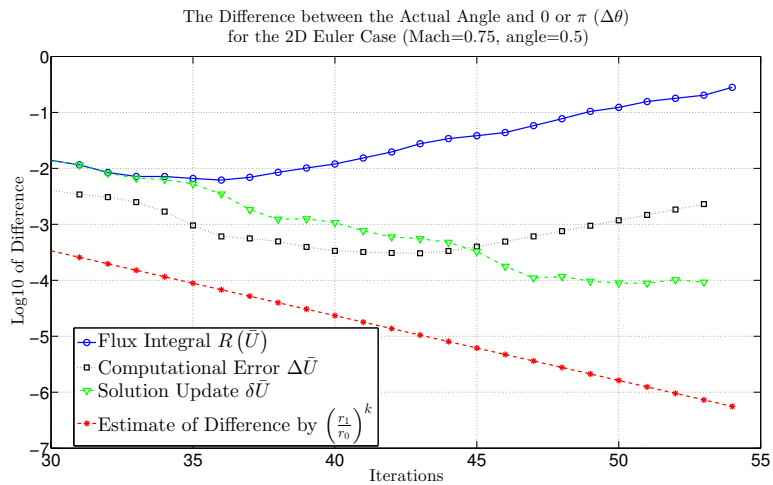


Figure 3.24: The difference between the actual angle and 0 or π ($\Delta\theta$) for the two dimensional Euler case with Mach=0.75, angle=0.5

3.5.2 Summary

Although the two dimensional Euler equations are nonlinear, the data and analysis above give us the following conclusions:

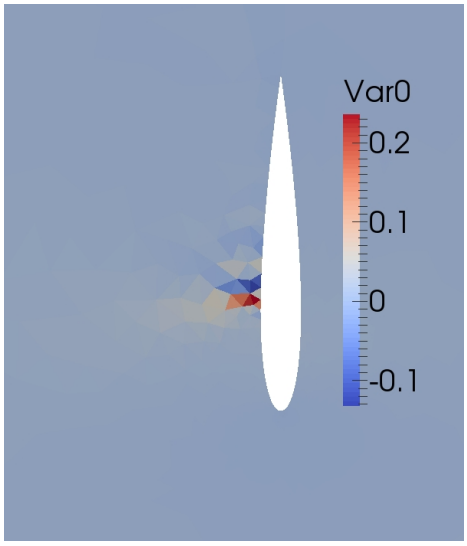
1. For the amplification factor: $\Delta\gamma_k$ is small for all three variables; there is a pattern of oscillation in $\Delta\gamma_k$ but the asymptotic rate matches $\frac{r_1}{r_0}$ for all three variables.

2. For the angle: $\Delta\theta_k$ is small for all three variables; there is a pattern of oscillation in $\Delta\theta_k$ but the asymptotic rate is consistent with $\frac{r_1}{r_0}$ for all three variables.

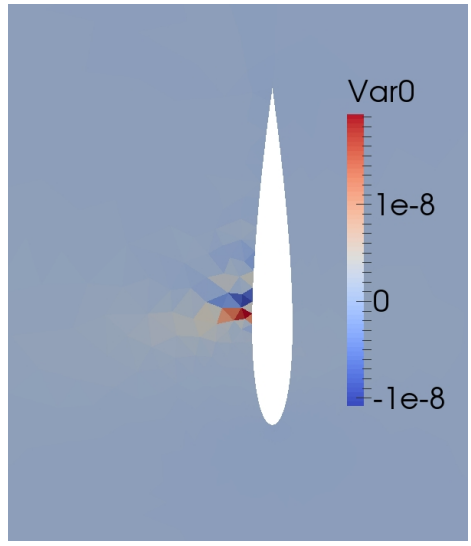
These conclusions demonstrate the methodology is valid for the two dimensional Euler equations where the largest eigenvalue is real for the associated Jacobian matrix.

3.6 Test Case No. 4 — Another Stable 2D Euler Equations

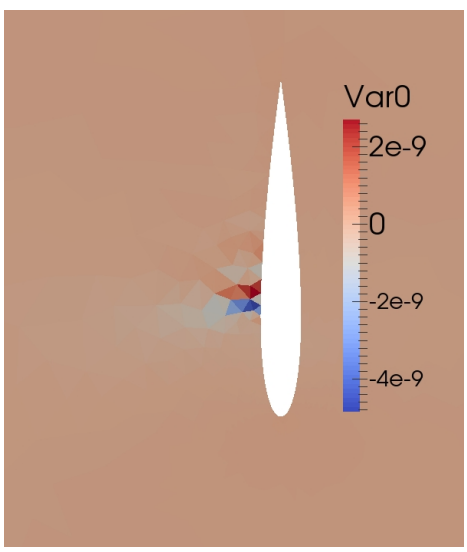
Different from Test Case No. 3, in this case, the Mach number is 1.05, and the attack angle is 0.5 degrees. Vertex-centered control volumes, and backward Euler



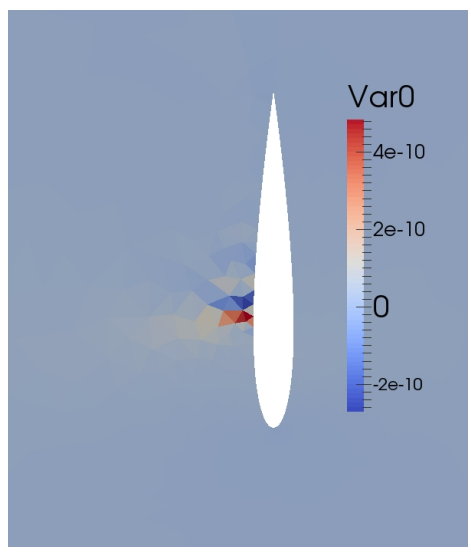
(a) The eigenvector associated with the largest eigenvalue



(b) The computational error at the 40th iteration



(c) The solution update at the 40th iteration



(d) The flux integral at the 40th iteration

Figure 3.25: Comparison for computational vectors and eigenvector for the 2D Euler case with Mach=0.75, angle=0.5 (density component, rotated for better visualization)

time-stepping with $dt = 5$ are used. Other settings are as same as those in Test Case No. 3.

The solution in terms of Mach field is presented in Figure (3.26), and the zoom of the Mach field around the airfoil is shown in Figure (3.27). The first twenty largest eigenvalues are plotted in Figure (3.28). The largest eigenvalue of the Jacobian matrix in this case are a conjugate complex pair, which is different from Test Case No. 3.

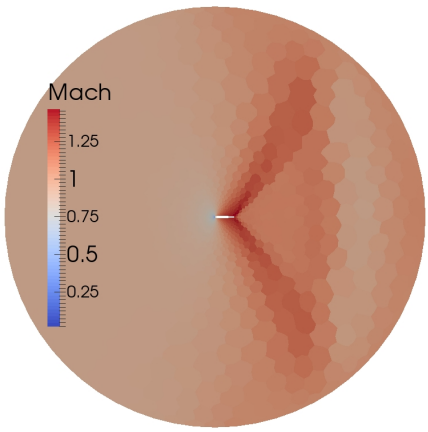


Figure 3.26: Mach field of 2D Euler case with Mach=1.05, angle=0.5 free stream

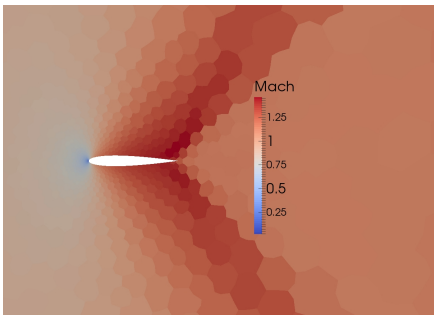


Figure 3.27: Zoom of the Mach field around the airfoil of 2D Euler case with Mach=1.05, angle=0.5 free stream

3.6.1 Numerical Results and Data Analysis

Figure (3.29) gives the visualization of the actual convergence histories. We can see from the figure that the asymptotic convergence rates of the three variables are consistent with the predicted rate from the eigenvalue. In Figure (3.30), amplification factors associated with two successive iterations are plotted, and we can see amplification factors oscillate around the norm of the largest eigenvalue. If we calculate the average amplification factors defined by Equation (3.6), we can see the results in Figure (3.31). Compared with Figure (3.30), the average amplification factor shows much less oscillation. The worst case has two digits precision.

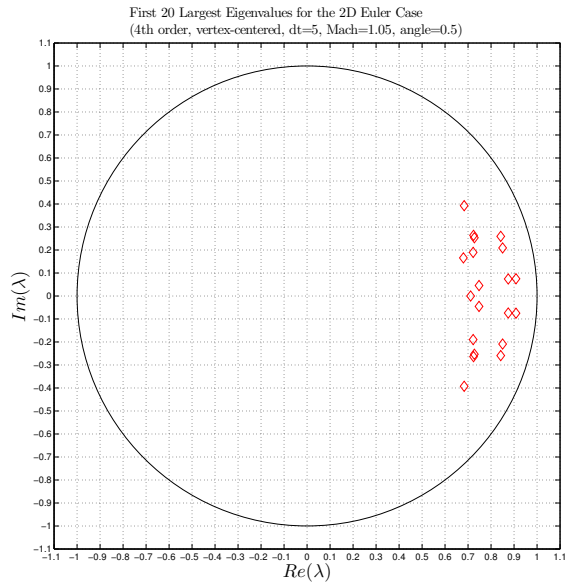


Figure 3.28: First twenty largest eigenvalues for the 2D Euler case with Mach=1.05, angle=0.5

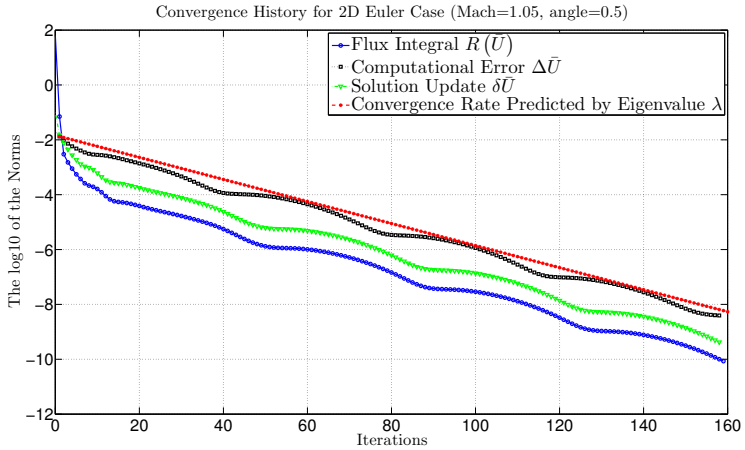


Figure 3.29: The convergence history for 2D Euler equations with Mach=1.05, angle=0.5

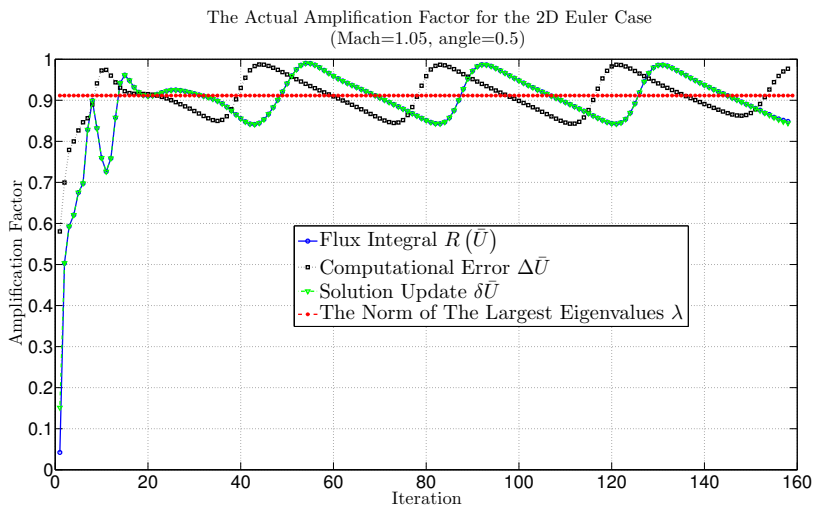


Figure 3.30: The amplification factors of two successive iterations for the 2D Euler case with Mach=1.05, angle=0.0

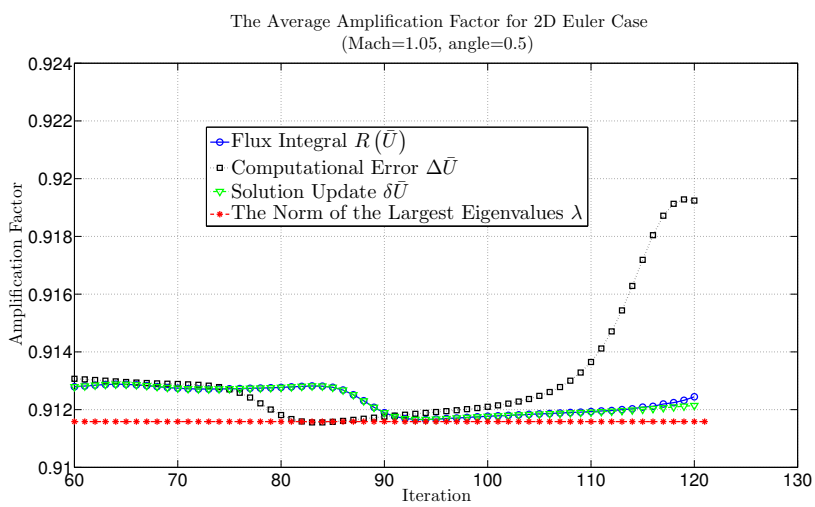


Figure 3.31: The average amplification factors for the two dimensional Euler case with Mach=1.05, angle=0.5

3.6.2 Summary

Though this case is nonlinear, we have similar conclusions as the advection-diffusion problem, which we do not repeat here.

3.7 Test Case No. 5 — Unstable 2D Euler Equations

The previous cases are all stable. In this section, we present a case of an unstable discretization system inherited from the 2D Euler equations. Different from Test Case No. 3, in this case, the Mach number is 1.2, and the attack angle is 0.0. Cell-centered control volumes, and backward Euler time stepping with $dt = 1$ are used. Other settings are as same as those in Test Case No. 3. This case is not stable for backward Euler time-stepping with some time-steps. However, in this thesis, we use the pseudo time-stepping method in [17] to reach the first order scheme solution, then use the first order scheme solution as the initial solution to implement time-stepping for the fourth order scheme with the same pseudo time-stepping method. For some of the cases in that the spatial discretization is not stable, this approach can reach the solution.

The solution in terms of Mach field is presented in Figure (3.32), and the zoom of the Mach field around the airfoil is shown in Figure (3.33). The first twenty largest eigenvalues are plotted in Figure (3.34). A zoom of the first two largest eigenvalues are also shown in Figure (3.35). Figure (3.36) shows first twenty right-most eigenvalues of the spatial Jacobian matrix. Figure (3.35) shows that two eigenvalues for the full discretization system are larger than one, which implies the system is not stable. In Figure (3.36), there are two eigenvalues in the right half plane, which implies that the semi-discretization is not stable.

3.7.1 Numerical Results and Data Analysis

The convergence histories in Figure (3.37) show that this case is not stable, which is consistent with the prediction both from the eigenvalues of the full Jacobian matrix and the spatial Jacobian matrix. We can also see the growth rates of the three variables coincide well with the rate predicted by the norm of the largest eigenvalue. Since $dt = 1$, the convergence history of the flux integral coincides with the convergence history of the solution update. Figure (3.38) presents the amplification factors associated with two successive iterations. We can see that after the 30th iteration, the amplification factors coincide with the norm of the largest eigenvalue.

As in previous cases, we calculate the difference between the actual amplification factors and the norm of the largest eigenvalue $\Delta\gamma$ to generate Figure (3.39).

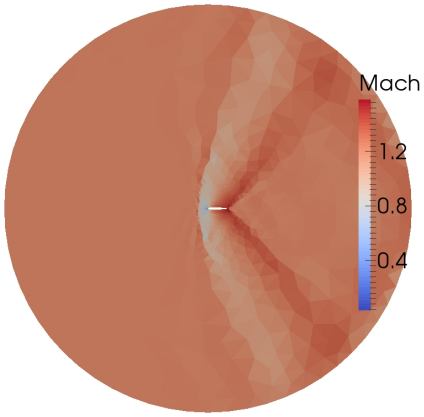


Figure 3.32: Mach field for the 2D Euler case with Mach=1.2, angle=0.0 free stream

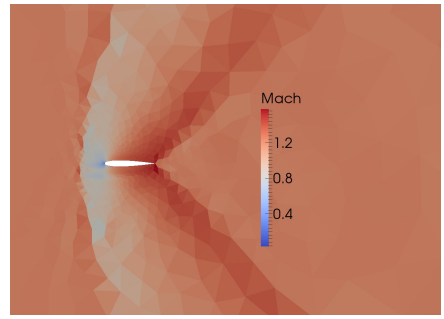


Figure 3.33: Zoom of the Mach field around the airfoil of the 2D Euler case with Mach=1.2, angle=0.0 free stream

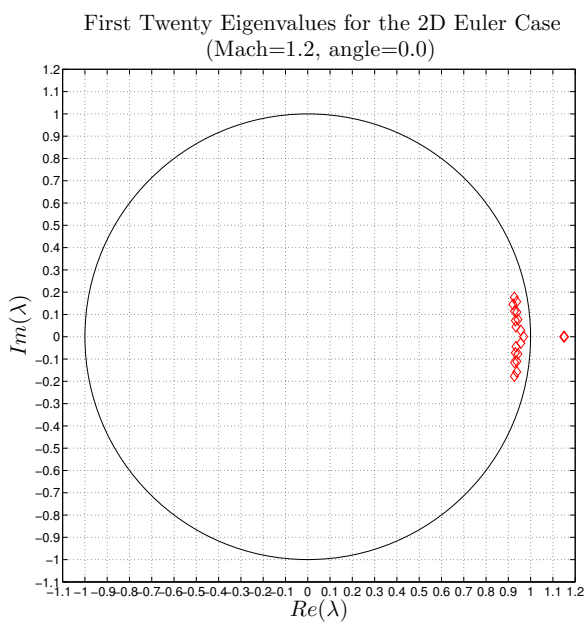


Figure 3.34: First twenty largest eigenvalues for the 2D Euler equations with Mach=1.2, angle=0.0 free stream

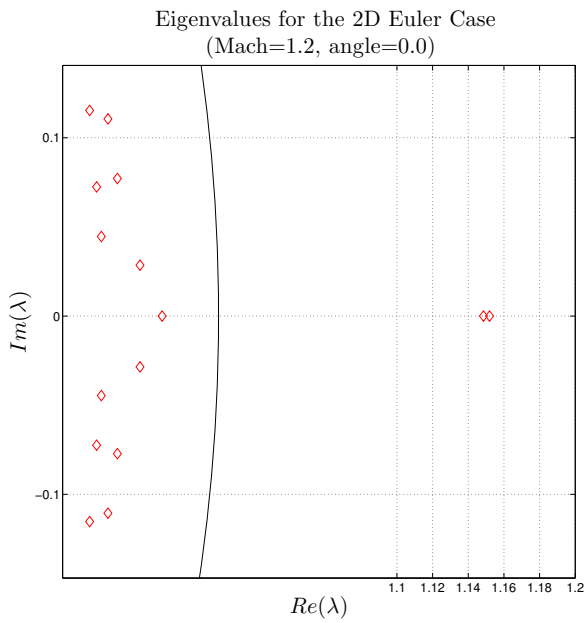


Figure 3.35: Zoom of the first two eigenvalues with for two dimensional Euler equations with Mach=1.2, angle=0.0 free stream

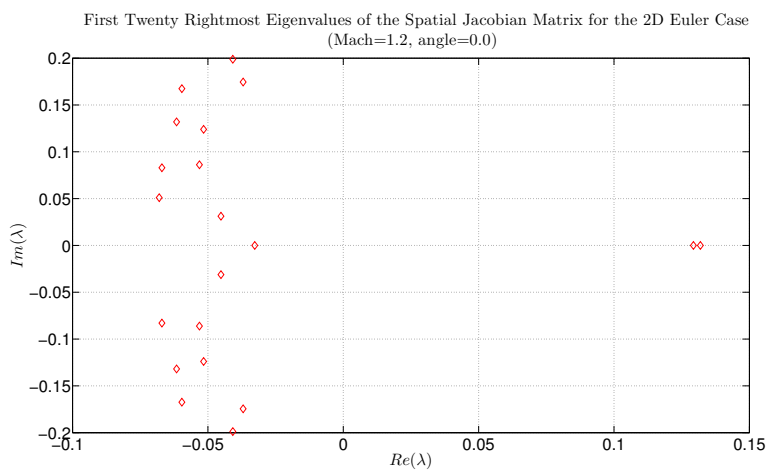


Figure 3.36: First twenty rightmost eigenvalues of the spatial Jacobian matrix for two dimensional Euler case with Mach=1.2, angle=0.0 free stream

We can see that the $\Delta\gamma$ is below 10^{-2} for all the three variables between iteration 30 and 74. Even the worst case in the figure is much smaller than $\left(\frac{r_1}{r_0}\right)^{k-1}$. For the decay rates, there is not a similarity between the computed variables and $\left(\frac{r_1}{r_0}\right)^{k-1}$.

For the angle, we can implement similar analysis and Figure (3.40) is generated. Compared with the amplification factors, the difference associated with the angle is larger but it is still much smaller than $\left(\frac{r_1}{r_0}\right)^{k}$. The decay rate of $\Delta\theta_k$ is not consistent with $\frac{r_1}{r_0}$ as well. That neither the decay rate of $\Delta\gamma_k$ nor the decay rate of $\Delta\theta$ are consistent with $\frac{r_1}{r_0}$ might be caused by the shock wave in the flow field. However, we still conclude the methodology is valid since both the values of $\Delta\gamma_k$ and $\Delta\theta_k$ are small.

Figure (3.41a) is the visualization of the density component of the eigenvector associated with the largest eigenvalue, Figure (3.41b) is the visualization of the vector of the computational error at 60th iteration in terms of density, Figure (3.41c) is the visualization of the vector of the solution update at 60th iteration in terms of density, and Figure (3.41d) is the visualization of the vector of flux integral at 60th iteration in terms of density. Comparing the shapes of the four vectors with each other, we can see the shapes are similar and the one associated with the flux integral is in the opposite direction.

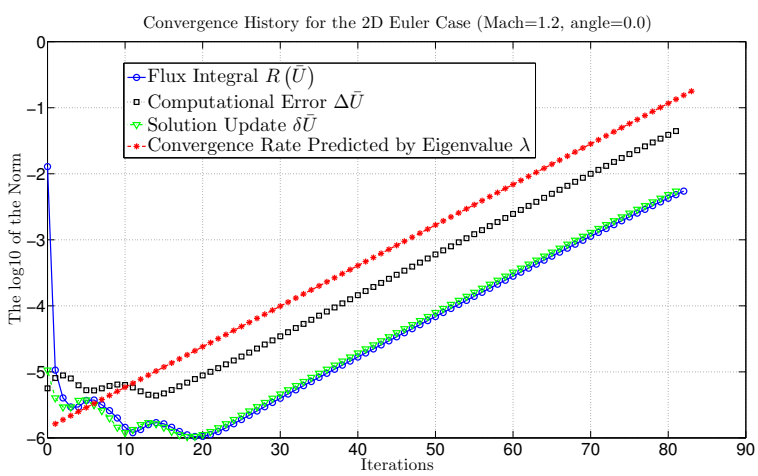


Figure 3.37: The convergence history for the 2D Euler case with Mach=1.2, angle=0.0

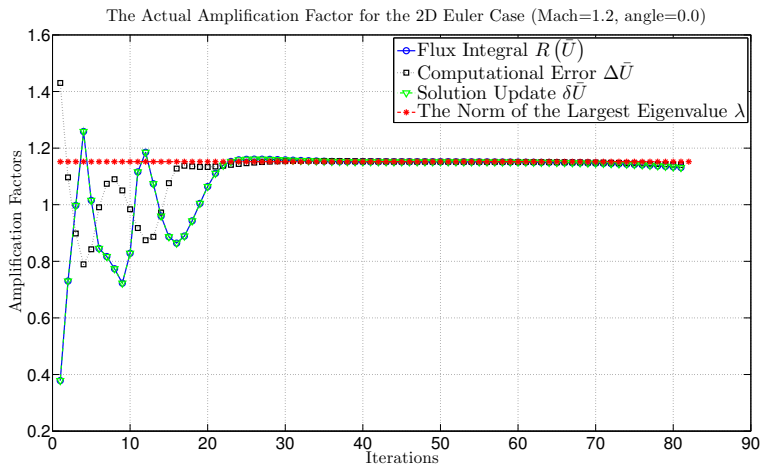


Figure 3.38: The amplification factor for the 2D Euler case with Mach=1.2, angle=0.0

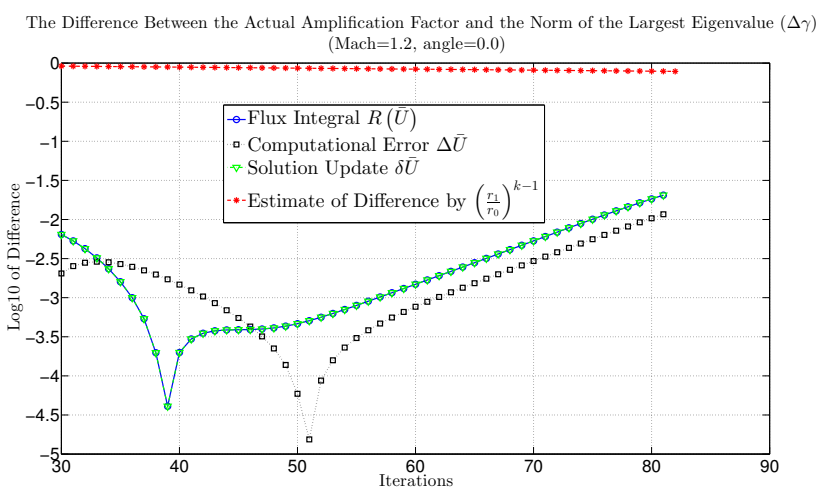


Figure 3.39: The difference between actual amplification factors and the norm of the largest eigenvalue ($\Delta\gamma$) for one unstable Euler equations case with Mach=1.2, angle=0.0

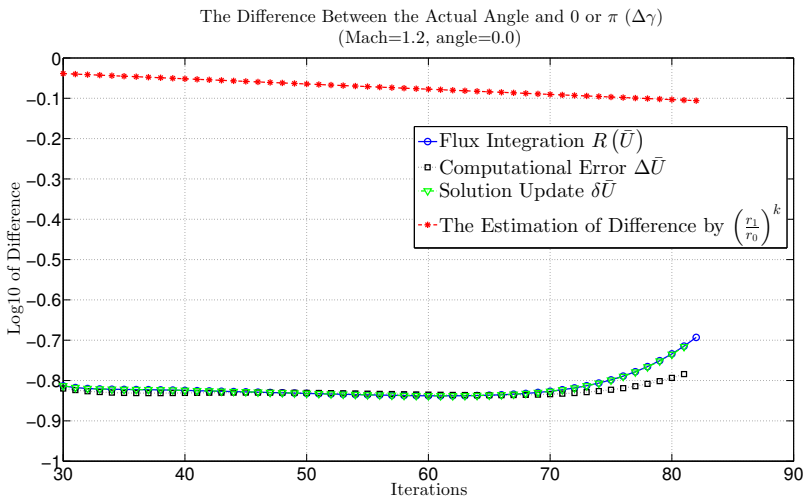


Figure 3.40: The difference between the actual angle and 0 or π ($\Delta\theta$) for one unstable Euler equations case with Mach=1.2, angle=0.0

3.7.2 Summary

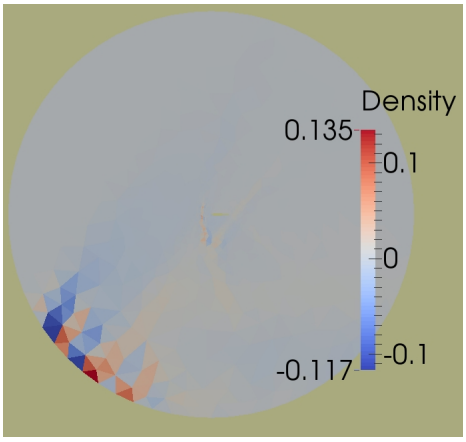
For this unstable nonlinear case, the following conclusions can be drawn from the data and analysis above:

1. For the amplification: $\Delta\gamma_k$ is small for all three variables but the decay rate does not coincide with $\frac{r_1}{r_0}$.
2. For the angle: $\Delta\theta_k$ is small for all three variables but the decay rate does not coincide with $\frac{r_1}{r_0}$.

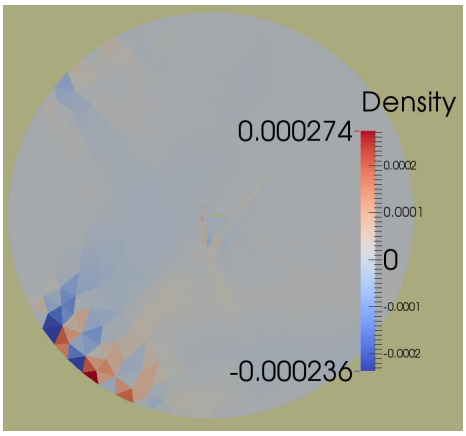
Though the decay rate does not coincide with $\frac{r_1}{r_0}$, we still think the methodology holds since the both the values of $\Delta\gamma_k$ and $\Delta\theta_k$ are small.

3.8 Summary of Stability Analysis

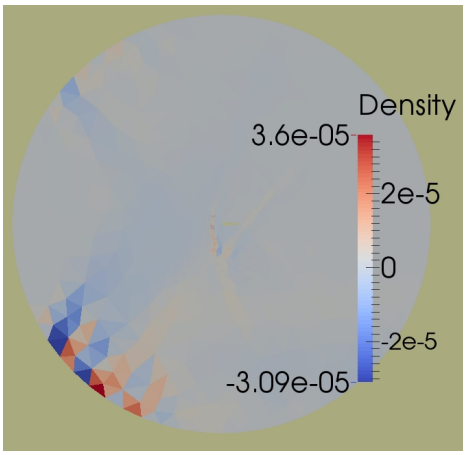
In Chapter 2, we developed and presented a new stability analysis methodology and constructed the corresponding stability condition. In this chapter, by testing linear problems and the nonlinear 2D Euler equations cases, we have verified the validity of the stability methodology.



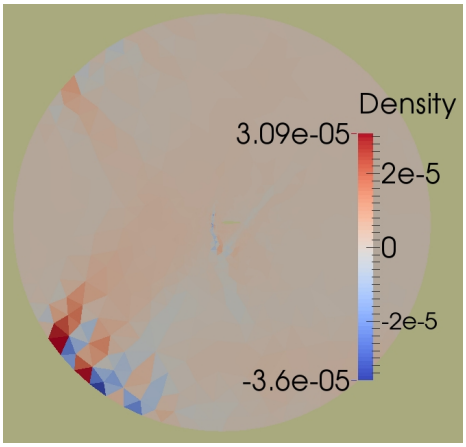
(a) The eigenvector associated with the largest eigenvalue



(b) Density component of the computation error at 60th iteration



(c) Density component of the solution update at the 60th iteration



(d) Density component of flux integral at the 60th iteration

Figure 3.41: Comparison among the computational vectors and the eigenvector associated with the largest eigenvalue (density component) for the 2D Euler case with Mach=1.2, angle=0.0.

Chapter 4

Stabilization at Fixed Point

In Chapter 2, we developed a stability analysis model, and constructed the associated stability condition in terms of eigenvalues. In addition, we also constructed a stability condition for the semi-discretization, which is independent of time-stepping. To have a stable ODE system, we need all the eigenvalues to lie in the left half plane. To stabilize an unstable ODE system, we need shift the eigenvalues in the right half plane into the left half plane. To reach this objective, we need change the Jacobian matrix. There are many ways to change the Jacobian matrix, but an easy and controllable way is to change the coordinates of the vertices of the mesh. The next question is how the changes of vertices' coordinates change the eigenvalues. Can we obtain a quantitative relation?

4.1 Finding the Derivative of the Eigenvalue

To determine how the eigenvalue changes quantitatively corresponding to a change of the matrix, we need the derivative of the eigenvalue. By the definition of right eigenvalue, we have

$$Ax = \lambda x. \tag{4.1}$$

For the left eigenvalue, we have

$$y^* A = \lambda y^*. \tag{4.2}$$

Taking the first derivative of both sides of Equation (4.1), we have

$$A^{'} x + A x' = \lambda^{'} x + \lambda x^{'}. \tag{4.3}$$

Multiplying both sides from the left by the left eigenvector yields

$$y^* A^{'} x + y^* A x' = \lambda^{'} y^* x + \lambda y^* x^{'}. \tag{4.4}$$

Multiplying Equation (4.2) from the right by the derivative of the right eigenvector gives

$$y^* A x' = \lambda y^* x'. \tag{4.5}$$

Substituting Equation (4.5) into Equation (4.4), we have

$$y^* A' x = \lambda' y^* x. \tag{4.6}$$

Dividing both sides by $y^* x$, we have

$$\frac{y^* A' x}{y^* x} = \lambda'. \tag{4.7}$$

By Equation (4.7), the derivatives of eigenvalues can be found from the derivative of the matrix, and the left and right eigenvectors. The earliest published derivation of Equation (4.6) might be Lancaster (1964) [11].

4.2 Prediction on the Difference of Eigenvalue under the Difference of a Matrix

We only take the derivative with respect to the coordinates of the mesh's vertices into consideration. We use finite differences to approximate the matrix's derivative with respect to the coordinate

$$A' = \frac{A(\zeta + \varepsilon) - A(\zeta)}{\varepsilon} + O(\varepsilon). \tag{4.8}$$

where ζ refers to the vector of the coordinates of vertexes, and ε is a perturbation of one coordinate of one vertex, which is a good starting point.[2] Substituting Equation (4.8) into Equation (4.7) yields

$$\frac{y^* \left(\frac{A(\zeta+\varepsilon)-A(\zeta)}{\varepsilon}\right) x}{y^* x} = \lambda'. \tag{4.9}$$

Equation (4.9) gives a way to approximate the eigenvalue's derivative approximately by finite differences. With the eigenvalue's derivative at hand, we can predict the eigenvalue of the new Jacobian matrix after mesh perturbation by a Taylor series, but only the first order derivative is taken into consideration:

$$\lambda_{new} = \lambda_{old} + \lambda'_{old} \varepsilon + O\left(\varepsilon^2\right). \tag{4.10}$$

Rearranging Equation (4.10), the difference of the eigenvalue can be predicted by

[2] The perturbation ε can be on one coordinate of one vertex or multiple coordinates of multiple vertices, but here the perturbation is only on one coordinate of one vertex.

$$\Delta\lambda = \lambda_{new} - \lambda_{old} = \lambda^{'}\varepsilon. \tag{4.11}$$

For a specific eigenvalue λ_i, where i is the index, we have

$$\Delta\lambda_i = \lambda_i^{new} - \lambda_i^{old} = \lambda_i^{'}\varepsilon. \tag{4.12}$$

Substituting Equation (4.9) into Equation (4.11) to have

$$\Delta\lambda = \lambda^{'}\varepsilon = \frac{y^*\left(A\left(\zeta+\varepsilon\right)-A\left(\zeta\right)\right)x}{y^*x} = \frac{y^*\Delta Ax}{y^*x}. \tag{4.13}$$

For a specific eigenvalue λ_i, we have

$$\frac{y_i^*\Delta Ax_i}{y_i^*x_i} = \Delta\lambda_i. \tag{4.14}$$

The difference of eigenvalue and the difference of the Jacobian matrix are related by Equation (4.14). For a certain perturbation to the matrix, there are two ways to calculate the difference of the eigenvalue. One is applying Equation (4.14) directly, i.e., the difference of the eigenvalue can be calculated by the difference of the matrix, and the left and right eigenvector. The other is obtaining the derivative of eigenvalue $\lambda^{'}$ by Equation (4.9), then gaining the difference of eigenvalue by Equation (4.12).

4.3 Numerical Verification of the Eigenvalue Difference Calculation Method

In this section, we are going to test the reliability of Equation (4.11), and determine the maximum perturbation ε to predict the difference $\Delta\lambda$ within an acceptable tolerance. We start directly on the two dimensional Euler equations rather than a linear problem. For the two dimensional Euler equations, if the coordinates are changed, the converged solution changes accordingly, and therefore, the Jacobian matrix changes as well. However, we do not take the change of the converged solution into consideration. We only take the rightmost eigenvalue λ_0 into consideration for testing.

For an irregular mesh, the length of the edges changes with location. Also, eigenvalues have different sensitivity to the coordinates at different locations. Generally, the solution changes rapidly where the mesh is dense. Therefore a small change of the vertex location in those regions can bring significant changes to the eigenvalues. On the other hand, the solution changes slowly where the mesh is sparse. A relatively large change of the vertex location does not necessarily change

the eigenvalues significantly. For clarity, we normalize the change of coordinate ε. We define the following length.

Definition 3. The unit length at a vertex is defined as 0.1 times the length of the shortest edge connected to the vertex: $L_u = 0.1\,|e_{shortest}|$.

We normalize the change of the coordinate ε by the unit length :

$$\varepsilon_s = \frac{\varepsilon}{L_u} \tag{4.15}$$

4.3.1 Procedure to Implement Tests

To implement numerical tests, we first calculate the derivative by using finite differences. The procedure to calculate the derivative is:

1. Calculate the rightmost eigenvalue, and the associated left and right eigenvector of the Jacobian matrix.

2. We use the central finite difference to approximate the derivative. Adjust one coordinate of one vertex by a small quantity ε_s to obtain a new Jacobian matrix $A(\zeta + \varepsilon_s)$. By Equation (4.14), we obtain the difference for the rightmost eigenvalue $\Delta\lambda_{0,1}$.

3. Adjust the same coordinate of the same vertex by an opposite quantity $-\varepsilon_s$ to obtain another new Jacobian matrix $A(\zeta - \varepsilon_s)$. By Equation (4.14), we obtain another difference for the rightmost eigenvalue $\Delta\lambda_{0,2}$.

4. Calculate the derivative with respect to the normalized length: $\lambda' = \frac{\Delta\lambda_{0,1} - \Delta\lambda_{0,2}}{2\varepsilon_s}$. We can construct a linear function of the difference of the rightmost eigenvalue $\Delta\lambda_0$ depending on the perturbation of one coordinate of one vertex:

$$\Delta\lambda_0 = \lambda_0^{'}\varepsilon_s \tag{4.16}$$

By Equation (4.16), we can predict the change of an eigenvalue arising from a perturbation. Besides the perturbations used for constructing Mapping (4.16), we implement other perturbations to verify the validity of Equation (4.16) and Equation (4.14). There are three ways to obtain the difference of the eigenvalue, of which one is by Equation (4.16), and another one is by Equation (4.14). The third one is by recomputing the eigenvalue of the corresponding new Jacobian matrix and therefore calculate the difference. We use the third one to evaluate the validity of the methods associated with Equation (4.14) and Equation (4.16).

For the test, each time we only perturb one coordinate of one vertex. For each coordinate, there are eight different perturbations. Two vertices are tested. The results are presented in the following subsection.

4.3.2 Numerical Results

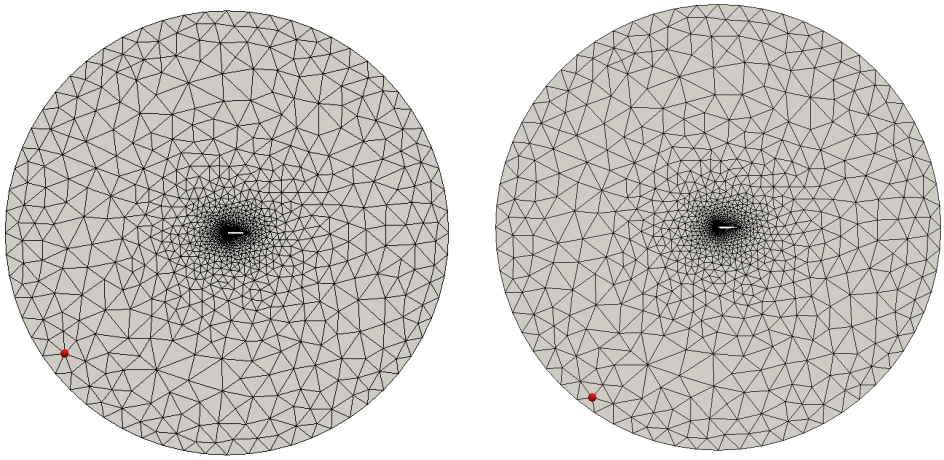
The red point in Figure (4.1a) is the vertex being perturbed. The number 683 is the index of the vertex. Figure (4.2) shows the change of the first rightmost eigenvalue under the perturbation of the x-coordinate of vertex 683. The unit for the x-axis is the L_u defined at the vertex 683. The perturbation associated with the length 0.01 and -0.01 are used to construct the linear map defined by Equation (4.16). Table (4.1) presents the relative error of the predicted change to the actual change associated with the largest four perturbations for both methods. The relative error of the predicted eigenvalue change is defined as

$$E_{\Delta\lambda_0} = \frac{\left|\Delta\lambda_0^{predicted} - \Delta\lambda_0^{actual}\right|}{\left|\Delta\lambda_0^{actual}\right|} \times 100\%. \tag{4.17}$$

From Table (4.1), we can see if the absolute value of the perturbation is not larger than one unit length, the prediction from linear Mapping (4.16) coincides with the actual change, i.e., the linearity is well preserved; (2) the prediction by Equation (4.14) matches the actual change as well for the absolute value of the perturbation not larger than one unit length. Even for a perturbation whose absolute value is 2 unit length, the relative error is still not too large. For this case, both the linear Mapping (4.16) and Equation (4.14) demonstrate a good prediction.

The red point in Figure (4.1b) is another point being perturbed, of which the index is 1417. Similar to vertex 683, we also generate Figure (4.3) and Table (4.2). For this case, we can see that if the absolute value of the change is within one unit length, the relative error of the prediction from both methods are acceptable. If the absolute value of the movement is as large as two unit length, the error is relatively large. However, if a movement within one unit length in absolute value cannot reach an objective, a larger movement like 1.5 or 2 unit length can also be taken into consideration.

From the two examples, we can see if a movement is not larger than one unit length (both positive and negative), both methods have shown good predictions. For a movement larger than one unit length, the prediction might not be sufficiently accurate, but preserves the correct trend. A movement larger than one unit length can be used if a smaller movement cannot reach the objective. We assume this conclusion applies to other eigenvalues, other vertices, and other coordinates. For a movement that simultaneously changes multiple coordinates, we assume superposition can be applied and do not confirm this by numerical tests.



(a) Vertex 683 (b) Vertex 1417

Figure 4.1: The perturbed vertices

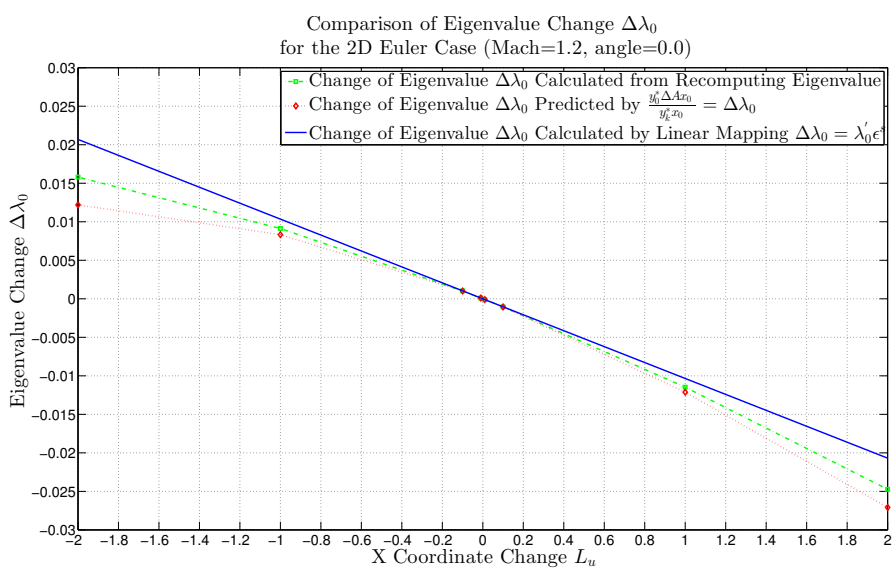


Figure 4.2: Change of eigenvalue when moving vertex 683 in the x-direction for the 2D Euler case with Mach=1.2, angle=0.0 free stream

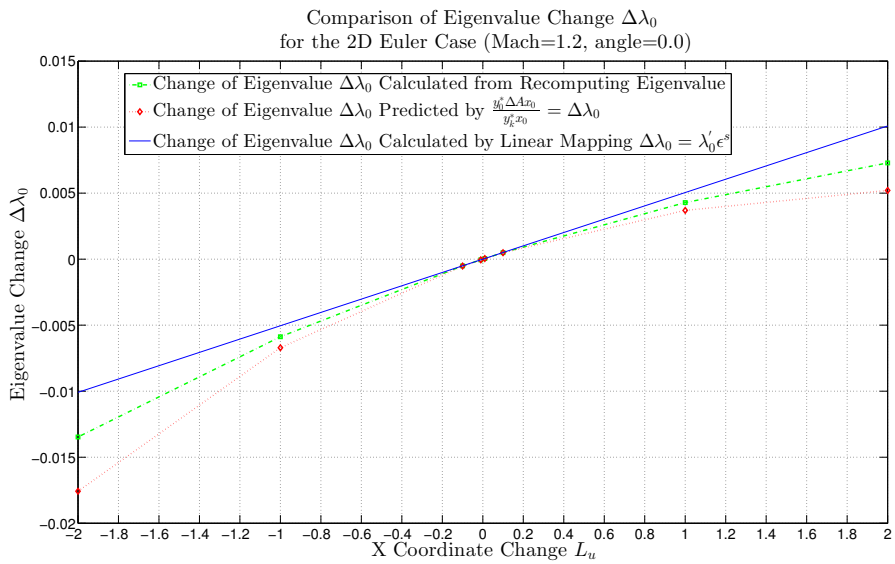


Figure 4.3: Change of eigenvalue when moving vertex 1417 in the x-direction for the 2D Euler case with Mach=1.2, angle=0.0 free stream

	$-2l_u$	$-L_u$	L_u	$2L_u$
Mapping (4.16)	31%	13%	10%	16%
Equation (4.14)	23%	9%	6%	9.5%

Table 4.1: Relative error of the prediction on the rightmost eigenvalue change for both methods under the movement of vertex 683

	$-2l_u$	$-L_u$	L_u	$2L_u$
Mapping (4.16)	25%	14%	18%	38%
Equation (4.14)	31%	14%	14%	29%

Table 4.2: Relative error for the prediction on the rightmost eigenvalue change for both methods under the movement of vertex 1417

4.4 Stabilization Methodology

From the last section, we know that if a coordinate's movement is around one unit length or even two unit lengths, the eigenvalue's change can be predicted well by the linear Mapping (4.16). For multiple-coordinate movement, we can apply superposition. To stabilize an unstable case, the eigenvalues on the right half plane need to be shifted into the left half plane and meanwhile, the originally negative eigenvalues should not be shifted to the right half plane.

4.4.1 Procedure of Systematic Stabilization

The next question is what is the adjustment for each vertex to shift the eigenvalue with positive real part into the left half plane. For each vertex, there are two coordinates, which we consider independently. Suppose m x-coordinates and n y-coordinates are adjusted. The adjustment for the x-coordinate is denoted by δx_i, and the adjustment for the y-coordinate is denoted by δy_j; as the index to list the coordinates to be moved, i ranges from zero to $m-1$ and j ranges from zero to $n-1$. Combine the adjustment for all coordinates of which the adjustment is not zero as a vector $\Delta\zeta = (\delta x_0, \delta x_1, \delta x_2, \ldots, \delta x_{m-1}, \delta y_0, \delta y_1, \ldots, \delta y_{n-1})$. To shift the eigenvalues with positive real part into the left half plane, we need to obtain the corresponding $\Delta\zeta$. Suppose the derivatives of the first several rightmost eigenvalues with respect to the coordinates of each vertex are obtained; we only need to consider the real part. For the coordinates to be adjusted, we can construct a matrix. Suppose we take the first l rightmost eigenvalues into consideration: $(\lambda_0, \lambda_1, \lambda_2, \ldots, \lambda_{l-1})$. For each eigenvalue λ_k, the derivative of its real part with respect to one normalized x-coordinate is $\frac{dRe(\lambda_k)}{dx_{s,i}}$,[3] and its derivative with respect to one normalized y-coordinate is $\frac{dRe(\lambda_k)}{dy_{s,j}}$. We construct a $l \times (m+n)$ matrix

$$\frac{dRe(\Lambda)}{d\zeta} = \begin{bmatrix} \frac{dRe(\lambda_0)}{dx_{s,0}} & \frac{dRe(\lambda_0)}{dx_{s,1}} & \cdots & \frac{dRe(\lambda_0)}{dx_{s,m-1}} & \frac{dRe(\lambda_0)}{dy_{s,0}} & \cdots & \frac{dRe(\lambda_0)}{dy_{s,n-1}} \\ \frac{dRe(\lambda_1)}{dx_{s,0}} & \frac{dRe(\lambda_1)}{dx_{s,1}} & \cdots & \frac{dRe(\lambda_1)}{dx_{s,m-1}} & \frac{dRe(\lambda_1)}{dy_{s,0}} & \cdots & \frac{dRe(\lambda_1)}{dy_{s,n-1}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{dRe(\lambda_{l-1})}{dx_{s,0}} & \frac{dRe(\lambda_{l-1})}{dx_{s,1}} & \cdots & \frac{dRe(\lambda_{l-1})}{dx_{s,m-1}} & \frac{dRe(\lambda_{l-1})}{dy_{s,0}} & \cdots & \frac{dRe(\lambda_{l-1})}{dy_{s,n-1}} \end{bmatrix}. \tag{4.18}$$

For convenience, we use D to denote matrix $\frac{dRe(\Lambda)}{d\zeta}$. For a vertex adjustment operation $\Delta\zeta$, the changes of the real parts of the eigenvalues will be approximately $Re(\Delta\lambda) = D\Delta\zeta$. In practice, $\Delta\zeta$ is unknown, and $Re(\Delta\lambda)$ is an objective that is set in advance such that all the eigenvalues are negative in real part:

[3] The coordinate is normalized by the unit length; so is the y-coordinate.

$$Re(\lambda) + Re(\Delta\lambda) < 0 \tag{4.19}$$

Therefore, we know D and $Re(\Delta\lambda)$, and $\Delta\zeta$ is unknown. $\Delta\zeta$ can be obtained by solving an optimization problem:

$$\min_{\Delta\zeta} \frac{1}{2}\Delta\zeta^{\intercal}\Delta\zeta \quad \text{such that} \begin{cases} D\Delta\zeta \leq Re(\Delta\lambda) \\ lb \leq \Delta\zeta \leq ub \end{cases} \tag{4.20}$$

We want to minimize the vertex adjustment since a larger adjustment means a larger divergence to the prediction of the eigenvalues' change and more chance to make the mesh invalid. lb and ub are used to bound the adjustment of vertices. Therefore, we summarize the procedure as:

1. Calculate the derivatives $\frac{dRe(\lambda_k)}{dx_{s,i}}$ and $\frac{dRe(\lambda_k)}{dy_{s,j}}$ for the eigenvalues of interest and all vertices in the mesh.
2. Set $Re(\Delta\lambda)$ such that Condition (4.19) is satisfied.
3. Solve the optimization problem of Equation (4.20) to obtain $\Delta\zeta$
4. Apply $\Delta\zeta$ to the mesh

4.5 Technical Results of Stabilization

In this subsection, we use the procedure above to stabilize two unstable two dimensional Euler equation cases.

4.5.1 Test Case No. 1

This case is the unstable one documented in Chapter 3 with Mach=1.2, angle=0.0 free stream and we do not repeat the description here. The derivatives $\frac{dRe(\lambda)}{d\varepsilon_s}$ are calculated for all the interior vertices and the first thirty rightmost eigenvalues. Here ε_s is the normalized x-coordinate or y-coordinate. We do not move the vertices at the boundary since this changes the domain and we set the derivatives associated with the boundary vertices zero. The derivative with respect to the x-coordinate or the y-coordinate for all vertices can be interpreted as a vector of the dimension equal to the number of the vertices. It is normal and common that we visualize the flow variables, for instance, the density, on the mesh. We can define a pseudo field variable similar to flow variables for visualization purpose, and we need the average value of this variable on the vertex centered control volume to

equal the derivative of the eigenvalue's real part with respect to the one of the normalized coordinates of this vertex: $\frac{dRe(\lambda)}{d\varepsilon_s}$. Therefore, if we visualize this variable on the mesh like other flow variables, the color of of each vertex-centered control volume represent the value of $\frac{dRe(\lambda)}{d\varepsilon_s}$. Other variables that can be mapped to the mesh can be visualized in this manner as well. As an example, Figure (4.4) shows the derivatives of the real part of the rightmost eigenvalue with respect to the x-coordinate of the vertex. We can see that most of the derivatives are close to zero.

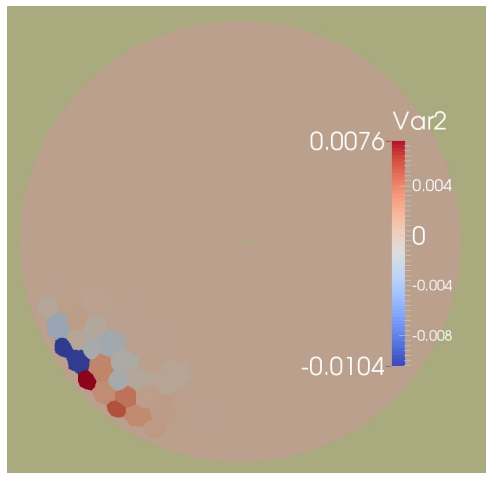


Figure 4.4: The derivative of the rightmost eigenvalue's real part with respect to the normalized x-coordinate $\frac{dRe(\lambda_0)}{dx_s}$ for the 2D Euler case with Mach=1.2, angle=0.0 free stream.

The optimization problem is solved by the function quadprog [15] in MATLAB [16]. The full description of the optimization problem that quadprog solves is described in [15] as:

$$\min_x \frac{1}{2}x^\intercal Hx + f^\intercal x \quad \text{such that} \begin{cases} A \cdot x \leq b \\ Aeq \cdot x = beq \\ lb \leq x \leq ub \end{cases} \tag{4.21}$$

Corresponding to the problem here, we set matrix H to be an identity matrix, matrix A to be matrix D, function f to be zero, Aeq and beq to be null, x to be $\Delta\zeta$ in Equation (4.20), b to be $Re(\Delta\lambda)$ in Equation (4.20), and the meanings of lb and ub in Equation (4.20) are same as those in Equation (4.21). Specifically, here we set lb to be a vector with all entries equal -1.5, and ub to be a vector with all entries equal 1.5, meaning the bound is 1.5 unit length in absolute value. For

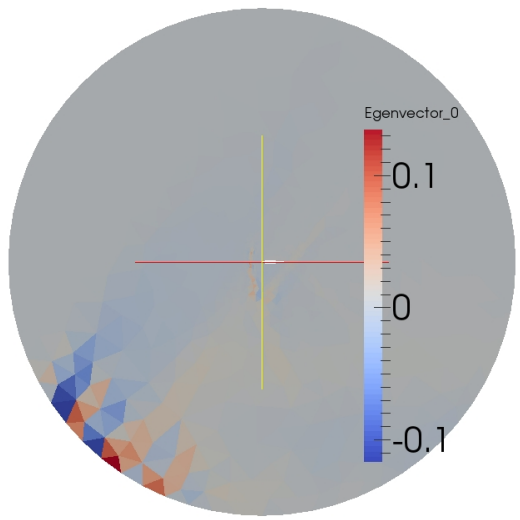


Figure 4.5: The eigenvector associated with the rightmost eigenvalue for the 2D Euler case with Mach=1.2, angle=0.0 free stream

$Re(\Delta\lambda)$, if the real part of the eigenvalue is larger than zero, we set the associated component of $Re(\Delta\lambda)$ to be -1.1 times of the real part; if the eigenvalue is smaller than zero, we set the associated component of $Re(\Delta\lambda)$ to be -0.1 times of the real part. The objective is to require the real part of the shifted eigenvalue of the original eigenvalue with positive real part be less than -0.1 times of the original real part and the real part of the shifted eigenvalue of the eigenvalue with negative real part be less than 0.9 times of the original real part. If this objective is satisfied, all the eigenvalues are negative in real part. We take the first 15 rightmost eigenvalues into consideration; for each coordinate, we have 15 $\frac{dRe(\lambda_i)}{d\varepsilon_s}$, i.e., i ranges from 0 to 14. For a specific eigenvalue, most of the derivative components are close to zero so that they cannot be used since a too large movement and too many coordinates are needed to shift the eigenvalue to a target value. We select those coordinates of which the derivatives are relatively large. Here the criterion to select one coordinate is that if among the 15 derivatives, one of the derivatives is larger than 0.0005 but smaller than 2,[4] the coordinate is chosen. Under this criterion, 98 coordinates among the 3216 coordinates of the mesh are selected. As a result of this, the size of matrix A is 15×98. However, the function quadprog needs matrix A be a square matrix. To remedy this, we extend the size to 98×98, and assign all of the extended entries to be zero. Accordingly, we extend the dimension of vector $Re(\Delta\lambda)$ from

[4]The reason why we need an upper bound is to avoid possible singularity.

15 to 98, and assign the extended entries the same value as the 15th entry.

After solving the optimization problem, we get the movement for the x-coordinates shown in Figure (4.19a) and the movement for the y-coordinates shown in Figure (4.19b). Both the original mesh and the new mesh after moving are shown in Figure (4.6) for comparison. The zoom of Figure (4.6) near the airfoil is shown in Figure (4.7), from which we can see the vertices near the airfoil are not moved. The new mesh is shown separately in Figure (4.8). We can see that the new mesh looks normal. Figure (4.9) shows the convergence history both for the time-stepping associated with the original mesh and the time-stepping associated with the new mesh. From the convergence history comparison, we can clearly see the original unstable time-stepping has been stabilized. Figure (4.10) shows the first twenty rightmost eigenvalues calculated on the converged solution after the mesh movement; compared with Figure (3.36), we can see that the original eigenvalues with positive real part have been shifted into left half plane. For those eigenvalues of which the original reals part are positive, the corresponding predictions $Re(\lambda)+Re(\Delta\lambda)$ from the optimization problem do not coincide well with the actual values of the updated eigenvalues. We do not study this further.

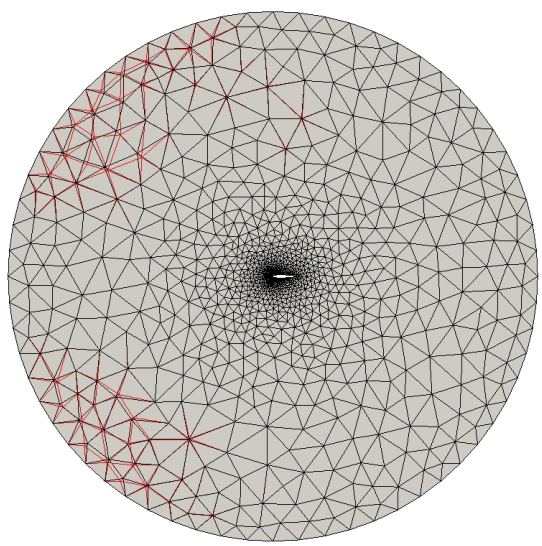


Figure 4.6: Mesh comparison for the stabilization for the 2D Euler case with Mach=1.2 and angle=0.0 (The red line is for the new mesh, and the black line is for the original mesh.)

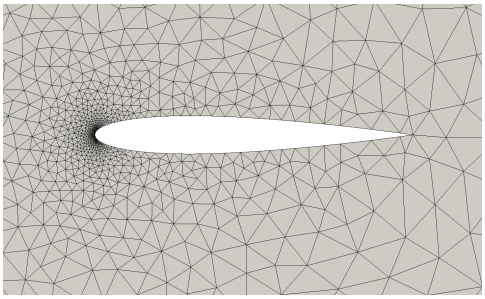


Figure 4.7: Zoom of Figure (4.6) near the airfoil

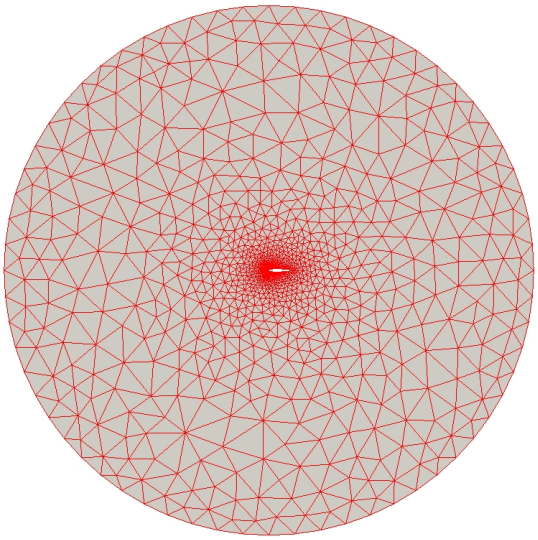


Figure 4.8: The new mesh arising from stabilization for the 2D Euler case with Mach=1.2 and angel=0.0

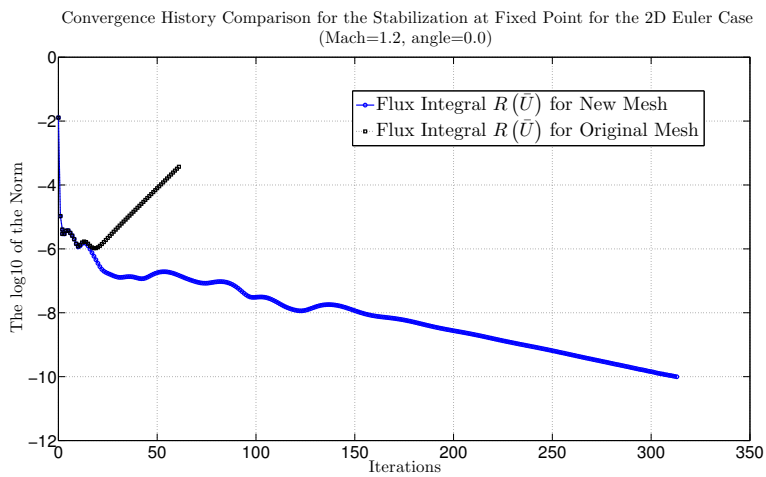


Figure 4.9: Convergence history comparison for the 2D Euler case associated with different meshes (Mach=1.2, angle=0.0)

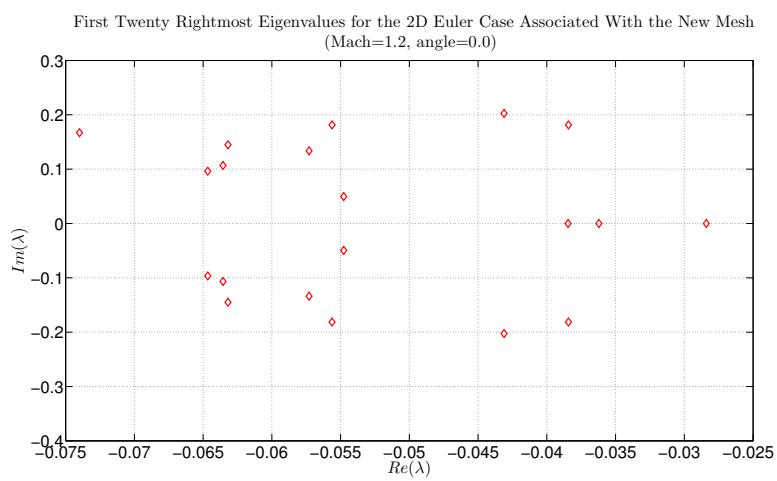


Figure 4.10: First twenty rightmost eigenvalues for the 2D Euler case associated with the new mesh (Mach=1.2, angle=0.0)

4.5.2 Test Case No. 2

It is also an unstable 2D Euler case. The parameters are all the same as the Test Case No. 1 except for that Mach number is 1.3 for the free stream. The solution in terms of Mach field is displayed in Figure (4.11). The density component of the eigenvector associated with the first rightmost eigenvalue is shown in Figure (4.20b). The density component of the eigenvector associated with the second rightmost eigenvalue is shown in Figure (4.20d). The first twenty rightmost eigenvalues calculated on the converged solution are plotted in Figure (4.12). The settings for the optimization problem are the same with the case with Mach=1.2, except that the first ten eigenvalues are taken into consideration.

The reason why we only take the first ten eigenvalues into consideration is because the derivatives associated with some eigenvalues from index 11 to index 15 are very large, and may arise from singularities, as shown in Figure (4.13). This can be studied in future work. After solving the optimization problem, we get the movement for the x-coordinate, which is shown in Figure (4.21a) and the movement for the y-coordinate, which is shown in Figure (4.21b). 134 coordinates among 3216 coordinates are changed. Both the original mesh and the new mesh after moving are shown in Figure (4.14) for comparison. The vertices near the airfoil are not moved. The new mesh is shown separately in Figure (4.15). We can see that the new mesh looks normal. The first twenty rightmost eigenvalues calculated on the converged solution on the new mesh are shown in Figure (4.16). Compared with Figure (4.12), we can see that the eigenvalues with positive real part have been shifted into the left half plane. The convergence history comparison is shown in Figure (4.17) and we can clearly see that the original unstable time-stepping has been stabilized.

4.6 The Region That Causes the Instability

In these two stabilization cases, the calculation of derivative $\frac{dRe(\lambda)}{d\varepsilon_s}$ goes through all the vertices of the mesh since we do not know which ones are to be selected in advance. However, in practice, only a small portion of the coordinates are needed. For the case with Mach=1.2, only 98 among 3216, i.e. 3.05%, coordinates are used for stabilization; for the case with Mach=1.3, only 134 among 3216, i.e. 4.17%, coordinates are used. Less than 5% of coordinates are actually used. The CPU time cost of mesh optimization for the 2D Euler case with Mach=1.2, angle=0.0 is presented in Table (4.3). In Table (4.3), Time-stepping means the time-stepping to reach the converged solution of the fourth order scheme. To calculate the derivatives, we need the eigenvalues, and the left and the right eigenvectors. Since we

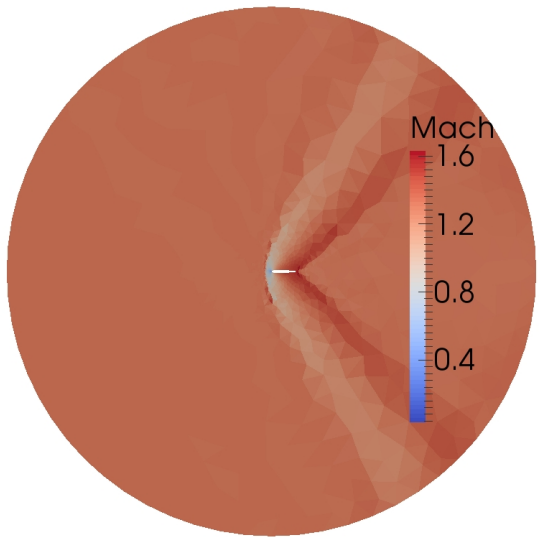


Figure 4.11: Mach field for the 2D Euler case with Mach=1.3, angle=0.0 free stream

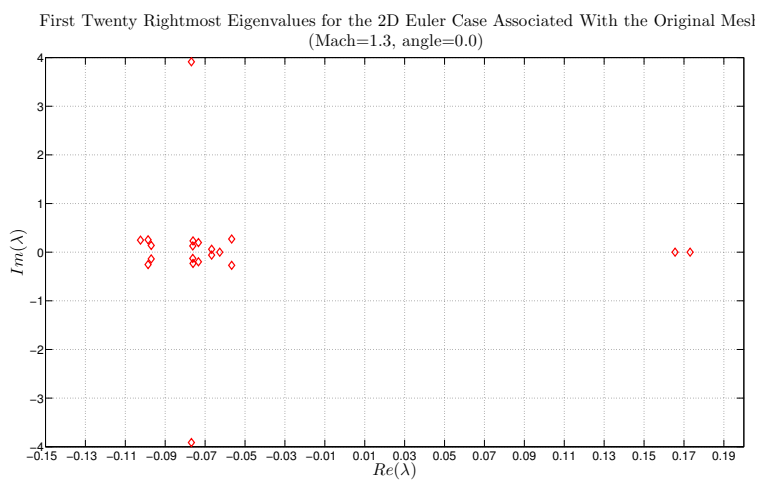


Figure 4.12: First twenty rightmost eigenvalues for the 2D Euler case associated with the original mesh (Mach=1.3, angle=0.0)

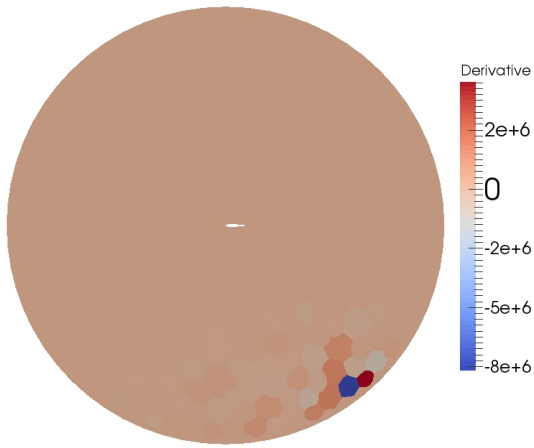


Figure 4.13: The derivative of the 14th rightmost eigenvalue's real part with respect to the normalized x-coordinate $\frac{dRe(\lambda_0)}{dx_s}$

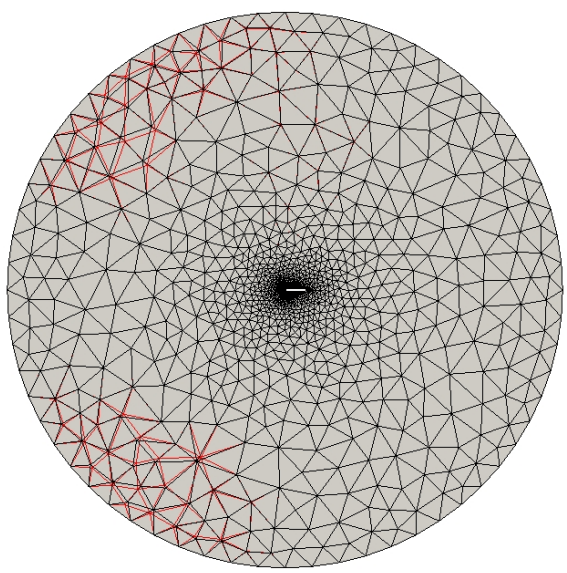


Figure 4.14: Mesh comparison for the stabilization for the 2D Euler case with Mach=1.3, angle=0.0 free stream (The black line is for the original mesh and the red line is for the new mesh.)

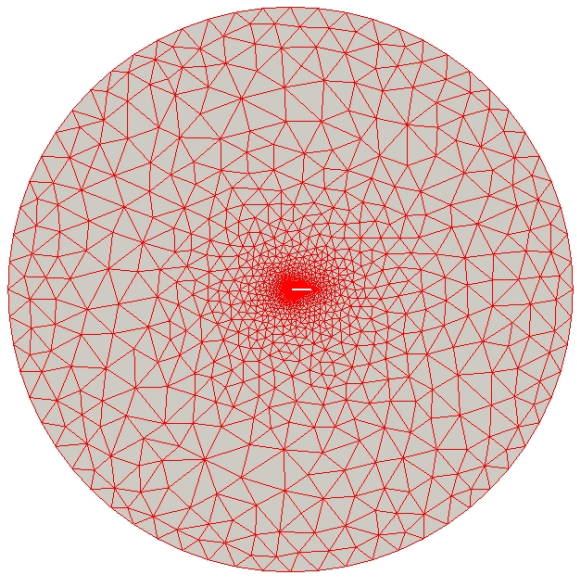


Figure 4.15: The new mesh after stabilization for the 2D Euler case with Mach=1.3, angle=0.0 free stream

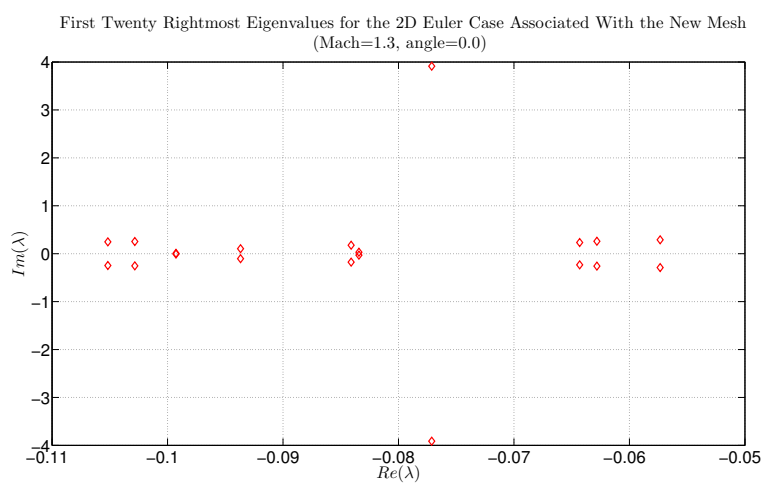


Figure 4.16: Rightmost eigenvalues for the 2D Euler case associated with the new mesh (Mach=1.3, angle=0.0)

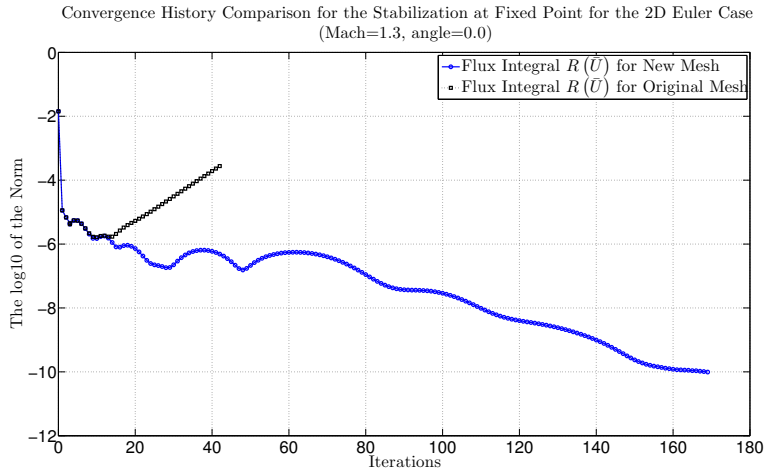


Figure 4.17: Convergence history comparison for the 2D Euler case for different meshes (Mach=1.3, angle=0.0)

cannot calculate the left and right eigenvectors at the same time, we transpose the Jacobian matrix to calculate the left eigenvectors. The 384s in the table includes both procedures to solve the right eigenvalue problem and the left eigenvalue problem. In the derivative calculation, for each coordinate, we calculate the first 30 rightmost eigenvalues' derivatives with respect to both coordinates of all the vertices. The CPU time used to solve the optimization problem is only 0.2s. In this case, 98% of CPU time is in derivative calculation. If we only calculate the derivatives for the coordinates which will be used in optimization, much CPU time can be saved. For instance, if we only calculate 10% of the coordinates, which are already more than two times the portion of the coordinates used in current cases, 88% of the computation time can be saved. The computation is executed in serial. If it is run in parallel, it takes much less time to complete the mesh optimization.

Items	Time (Second)
Time-Stepping	42
Eigenvalue/Eigenvector Calculation	384
Derivative Calculation	21,748

Table 4.3: Time cost for mesh optimization of the 2D Euler case with Mach=1.2

The question is how can we know that which coordinates will be used and which ones will not be used. We can see Figure (4.4) shows a similar nonzero

pattern as Figure (4.5), suggesting that there might be some relation between them. Usually the entries of an eigenvector of the Jacobian matrix arising from the spatial discretization of flow problems are not uniform in absolute value. In fact, the distribution of the entries in terms of absolute value are so far from being uniform that most of the entries are small and only a small part of the entries are much larger than the rest. Combine this fact with Equation (4.7) and Equation (4.14), and we can conclude that only a small part of the vertices of the mesh have a significant influence on the sensitivity of the eigenvalues. These are the vertices which lie in and around the area with large value in the visualization of the associated eigenvector. We examine more cases. The derivatives associated with the second rightmost eigenvalue shown in Figure (4.18a) show a similar nonzero pattern as the eigenvector associated with the second rightmost eigenvalue shown in Figure (4.18b), except the area around the airfoil. For the 2D case with Mach=1.3, we can see that the derivatives of the first rightmost eigenvalue's real part, shown in Figure (4.20a), show a similar nonzero pattern as the real part of the eigenvector associated with the first rightmost eigenvalue, shown in Figure (4.20b); the derivatives of the second rightmost eigenvalue's real part, shown in Figure (4.20c), show a similar nonzero pattern as the real part of the eigenvector associated with the second rightmost eigenvalue,[5] visualized in Figure (4.20d). The nonzero pattern of the derivative of the eigenvalue with respect to the y-coordinate is similar to the derivative of the eigenvalue with respect to the x-coordinate. From the analysis and numerical results, we conclude that the nonzero pattern of the eigenvalue's derivatives with respect to the vertices (x-coordinates and y-coordinates) coincides with the nonzero pattern of the associated eigenvectors.

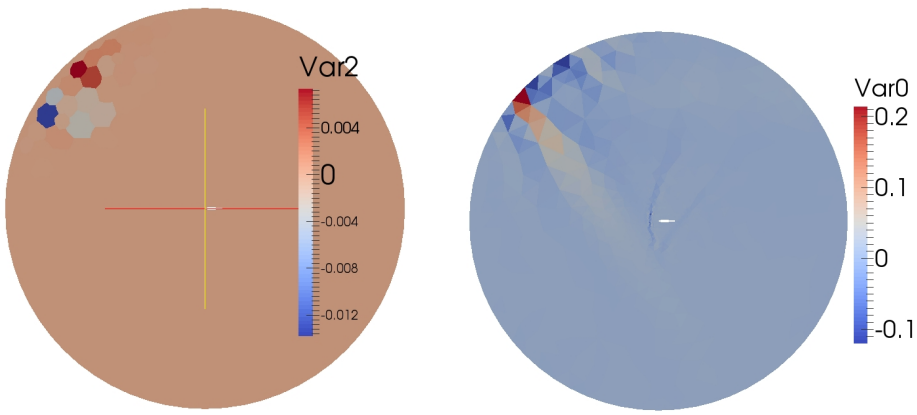
Since we primarily move those vertices whose associated derivatives are larger, we expect the nonzero patterns of the movement both for x-coordinate and y-coordinate to be similar to the superposition of the nonzero pattern of the associated eigenvectors. For the 2D Euler case with Mach=1.2, the x-coordinates' movement is shown in Figure (4.19a) and the y-coordinates' movement is shown in Figure (4.19b), both of which coincide with the superposition of the nonzero pattern of the eigenvector associated with the first rightmost eigenvalue, shown in Figure (4.5) and the eigenvector associated with the second rightmost eigenvalue, shown in Figure (4.18b).[6] For the 2D Euler case with Mach=1.3, from Figure (4.20b), Figure (4.20d), Figure (4.21a), and Figure (4.21b), we can see the same conclusion. Therefore, the areas with relatively large entries of the eigenvectors

[5] In this thesis, the unstable eigenvalues are all real. Therefore, the real part of the eigenvector is equivalent to the eigenvector.

[6] We take the first 15 eigenvalues into consideration. However, the first two rightmost eigenvalues are positive in real part and are primary taken to consideration. Therefore, the contribution of the rest eigenvectors to the pattern of the movement might be trivial.

associated with the unstable eigenvalues can be interpreted to cause the instability, since if we move those vertices, the unstable eigenvalues can be shifted to be stable ones.

As a result, we may just need to consider the vertices that are associated with the relatively large entries of the eigenvectors of interest.[7] We do not take this into further consideration, though we suggest it be done in future work.



(a) The derivative of the second rightmost eigenvalue's real part with respect to the x-coordinate

(b) The eigenvector associated with the second rightmost eigenvalue

Figure 4.18: The nonzero pattern comparison among the derivatives and eigenvectors for the 2D Euler case with Mach=1.2, angle=0.0 free stream

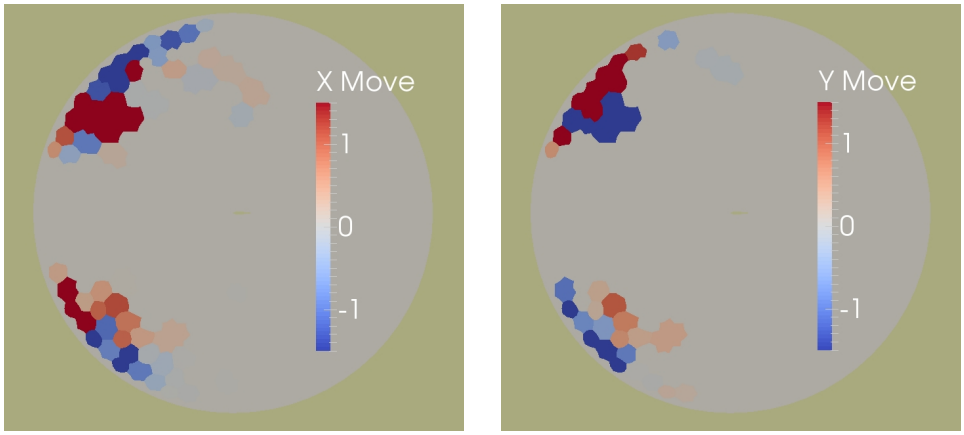
4.7 Summary

From this chapter, we have the following conclusions:

1. The changes in eigenvalues of the Jacobian matrix caused by moving the vertices of the mesh can be predicted quantitatively.

2. The unstable eigenvalues of the Jacobian matrix can be shifted to be stable eigenvalues by solving an associated optimization problem for a perturbation of the mesh.

[7] Here the eigenvectors of interest are referring to the eigenvectors associated with the eigenvalues being taken into consideration. Maybe we only need to consider the unstable eigenvalues, though in practice we take the first 15 or 10 rightmost eigenvalues into consideration.

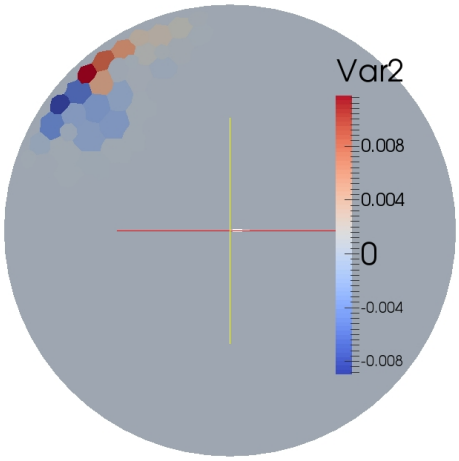


(a) X-coordinate movement

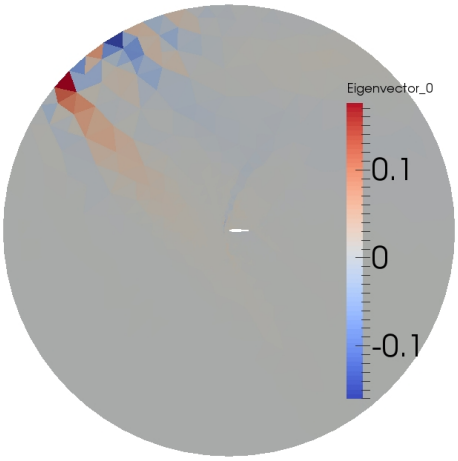
(b) Y-coordinate movement

Figure 4.19: The coordinates' movement to stabilize the unstable 2D Euler case with Mach=1.2, angle=0.0 free stream

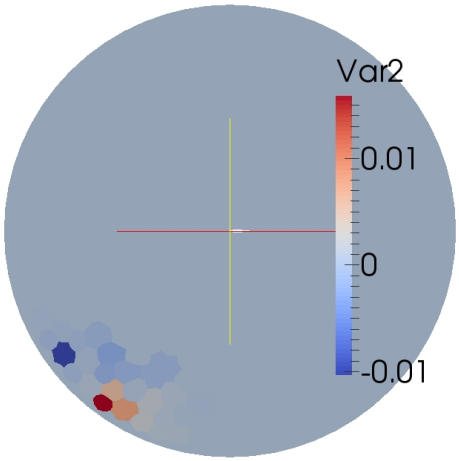
3. Only a small area of the whole mesh causes instability. If we only calculate the derivatives associated with the area primarily contributing to the instability, much computation time can be saved.



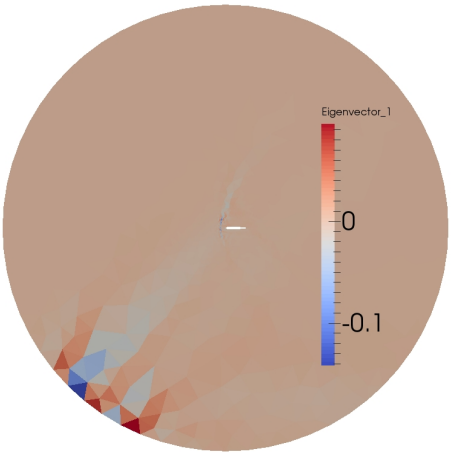
(a) The derivative of the first rightmost eigenvalue's real part with respect to the x-coordinate



(b) The density component of the eigenvector associated with the first rightmost eigenvalue

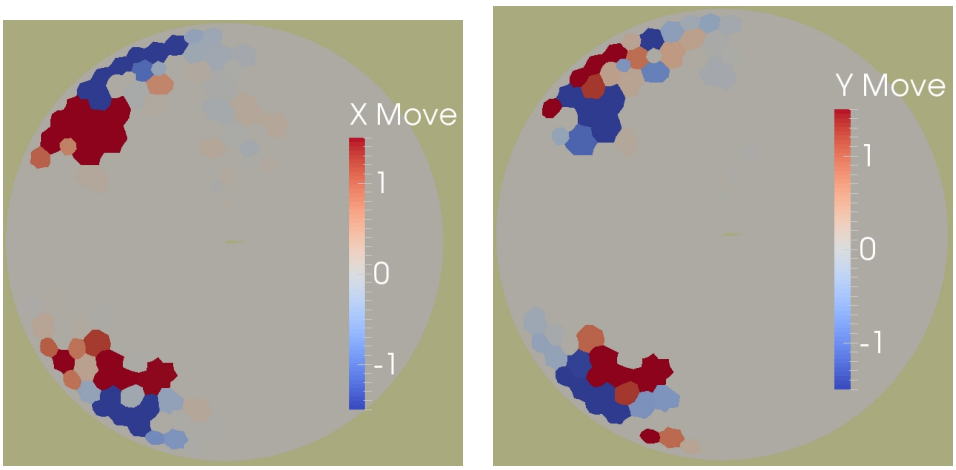


(c) The derivative of the second rightmost eigenvalue's real part with respect to the x-coordinate



(d) The density component of the eigenvector associated with the second rightmost eigenvalue

Figure 4.20: Comparison among the derivatives of eigenvalues and the eigenvectors for the 2D Euler case with Mach=1.3, angle=0.0 free steam



(a) X-coordinate movement (b) Y-coordinate movement

Figure 4.21: The coordinates' movement to stabilize the unstable 2D Euler case with Mach=1.3, angle=0.0 free stream

Chapter 5

Stabilization at Non-fixed Point

In Chapter 2, we developed a stability analysis model and its validity is verified by numerical tests in Chapter 3. This model is based on the fixed point of the time-stepping. In Chapter 4, we developed a method to stabilize unstable time-steppings by shifting the unstable eigenvalues into the stable region. This stabilization method is also applied at the fixed point. Fixed point analysis and stabilization at the fixed point can provide rich information about the system. However, to implement fixed point analysis, the fixed point should be reached. For some cases, the fixed point might not be feasible. In this chapter, we try to stabilize the unstable time-stepping at a non-fixed point.

Typically, the first order scheme does not have instability issues. Higher order schemes like the second, third, fourth order schemes, etc., have instability issues under some circumstances. To have a good initial solution, we usually set the first order solution as the initial solution for a high order scheme time-stepping. In this chapter, we try to stabilize high order schemes at the initial solution, which is the converged solution of the first order scheme.

We use the same procedure as the stabilization at the fixed point, but here we implement the stabilization at the initial solution, which is the converged solution of the first order scheme.

5.1 Test Case No. 1 — the Unstable 2D Euler Case with Mach=1.2, Angle=0.0 Free Stream

The boundary conditions, the spatial discretization, and time-stepping are as same as those in the previous 2D Euler case with Mach=1.2, angle=0.0 free stream in Chapter 4. Different from the stabilization in Chapter 4, in this chapter, we try to stabilize this unstable fourth order scheme at the initial solution, which is the fixed point of the first order scheme time-stepping. The first twenty rightmost eigenvalues of the Jacobian matrix at the initial solution are shown in Figure (5.1). The eigenvector associated with the rightmost eigenvalue is shown in Figure (5.2a). The eigenvector associated with the second rightmost eigenvalue is shown in Figure (5.2c). The derivative $\frac{dRe(\lambda_0)}{dx_s}$ is shown in Figure (5.2b); the derivative $\frac{dRe(\lambda_1)}{dx_s}$

is shown in Figure (5.2d); here λ_0 means the rightmost eigenvalue; λ_1 means the second rightmost eigenvalue.

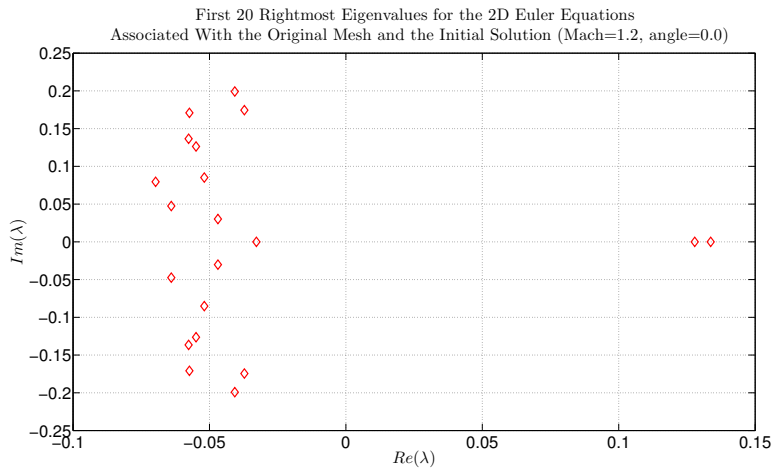
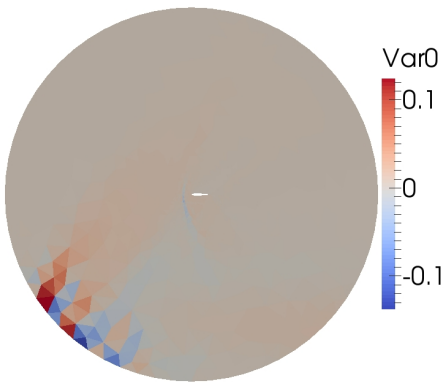


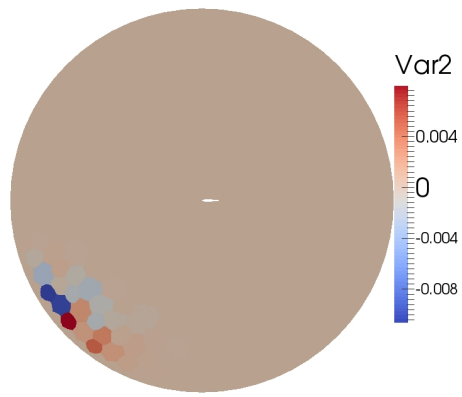
Figure 5.1: First twenty rightmost eigenvalues of the Jacobian matrix of the fourth order scheme at the initial solution for the 2D Euler case with Mach=1.2, angle=0.0 free stream

The settings for the optimization problem are as same as those for the 2D Euler case with Mach=1.2 in Chapter 4. After solving the optimization problem, the change in the mesh is obtained. The change in the x-coordinates is shown in Figure (5.3a) and the change in the y-coordinates is shown in Figure (5.3b). The original mesh and the new mesh are shown together in Figure (5.4) for comparison. The new mesh is shown in Figure (5.5). The new mesh looks normal though the largest movement for a single coordinate is 1.5 unit length.

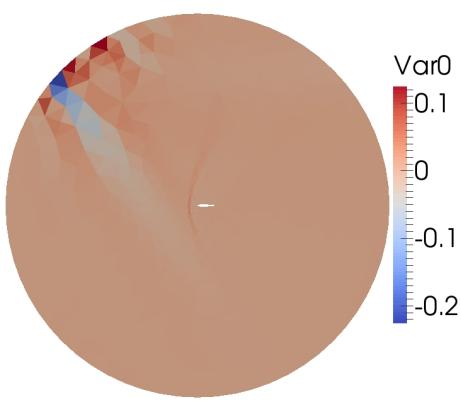
Once we obtain the updated mesh, we implement time-stepping from the initial solution, i.e., the solution of the first order scheme. Both the convergence histories in terms of flux integral for the time-stepping associated with the original mesh and for the time-stepping associated with the new mesh are shown in Figure (5.6), from which we can see that the unstable time-stepping has been stabilized. The first twenty rightmost eigenvalues of the Jacobian matrix calculated on the converged solution on the new mesh are shown in Figure (5.7). The eigenvalues are all negative in real part, while there are unstable eigenvalues in the spectrum associated with the original mesh, which is shown in Figure (3.36).



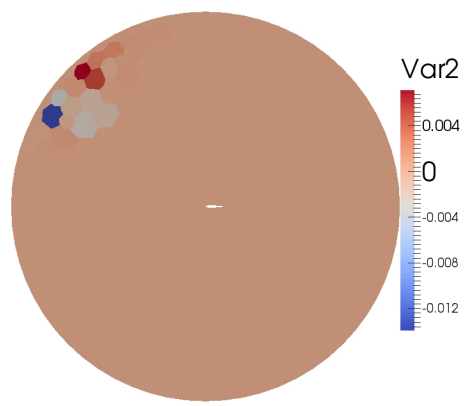
(a) The density component's eigenvector associated with the first rightmost eigenvalue of Jacobian matrix of the fourth order scheme at the initial solution



(b) The derivatives of the first rightmost eigenvalue's real part with respect to the normalized x-coordinate: $\frac{dRe(\lambda_0)}{dx_s}$

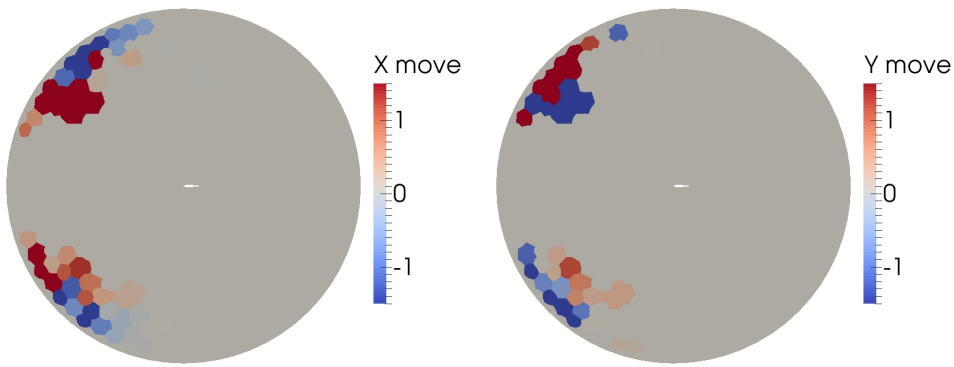


(c) The density component's eigenvector associated with the second rightmost eigenvalue of the Jacobian matrix of the fourth order scheme at the initial solution



(d) The derivatives of the second rightmost eigenvalue's real part with respect to the normalized x-coordinate: $\frac{dRe(\lambda_1)}{dx_s}$

Figure 5.2: Eigenvectors and the derivatives for the 2D Euler case with Mach=1.2, angle=0.0 free stream



(a) X-coordinate change (b) Y-coordinate change

Figure 5.3: The normalized coordinate's change for the non-fixed point stabilization for the 2D Euler case with Mach=1.2, angle=0.0 free stream

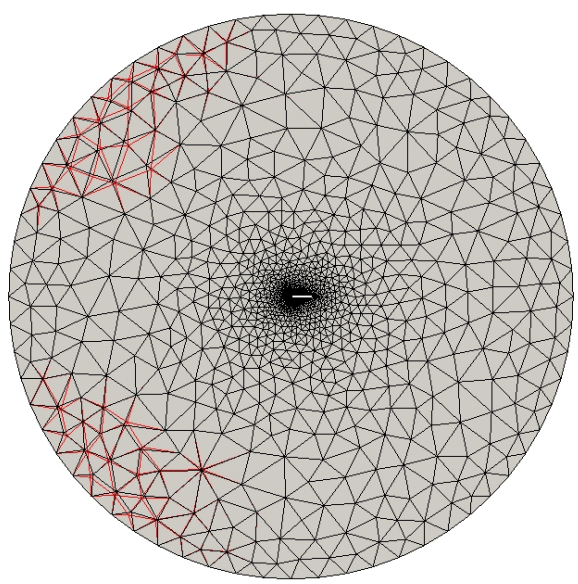


Figure 5.4: Mesh comparison for the stabilization at the initial solution for the 2D Euler case with Mach=1.2, angle=0.0 (The red line is for the new mesh, and the black line is for the original mesh)

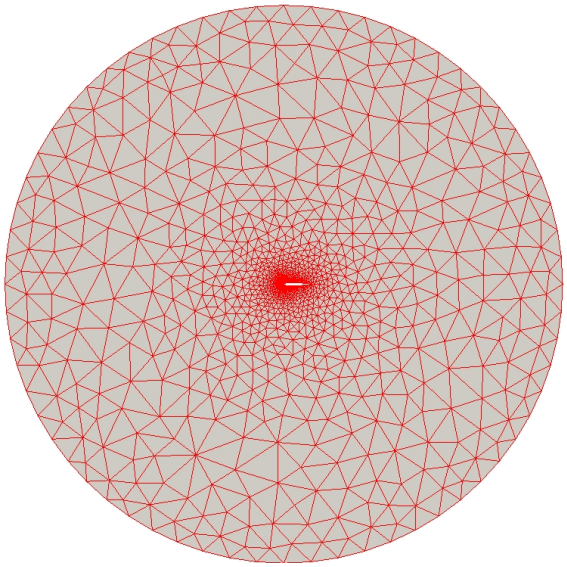


Figure 5.5: New mesh for the stabilization at the initial solution for the 2D Euler case with Mach=1.2, angle=0.0 free stream

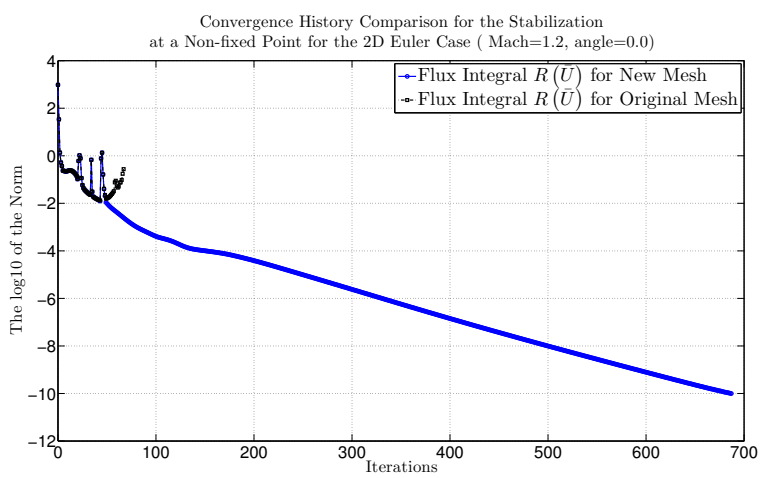


Figure 5.6: Convergence history comparison for the stabilization at a non-fixed point for the 2D Euler case (Mach=1.2, angle=0.0)

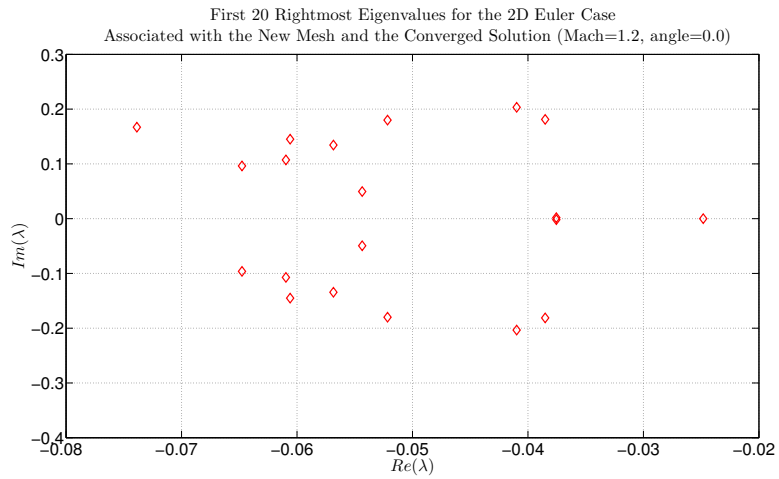


Figure 5.7: First twenty rightmost eigenvalues of the Jacobian matrix on the converged solution of the fourth order scheme on the new mesh for the 2D Euler case with Mach=1.2, angle=0.0 free stream

5.2 Test Case No. 2 — the Unstable 2D Euler Case with Mach=1.3, Angle=0.0 Free Stream

The boundary conditions, the spatial discretization, and time-stepping are as same as those in the previous 2D Euler case with Mach=1.3, angle=0.0 free stream in Chapter 4. We try to stabilize this unstable case at the initial solution, which is the solution of the first order scheme. The first twenty rightmost eigenvalues of the Jacobian matrix at the initial solution are shown in Figure (5.8). The eigenvector associated with the rightmost eigenvalue is shown in Figure (5.9a). The eigenvector associated with the second rightmost eigenvalue is shown in Figure (5.9c). The derivatives $\frac{dRe(\lambda_0)}{dx_s}$ are shown in Figure (5.9b); The derivatives $\frac{dRe(\lambda_1)}{dx_s}$ are shown in Figure (5.9d); here λ_0 means the rightmost eigenvalue; λ_1 means the second rightmost eigenvalue.

For the associated optimization problem, we only take the first four rightmost eigenvalues into consideration.[8] The criterion to select a coordinate is that there

[8]If we take more eigenvalues into consideration, for instance, first ten rightmost eigenvalues, the derivatives $\frac{dRe(\lambda)}{dx_s}$ and $\frac{dRe(\lambda)}{dy_s}$ for some eigenvalues may have singularity issues. The singularity may arise from a shock wave, or the solution may be non-physical since we do not employ any technique to guarantee the computational solution is valid in physical meaning. Blending stabilization with a technique to guarantee the computational solution is physically valid is a suggested topic of future work.

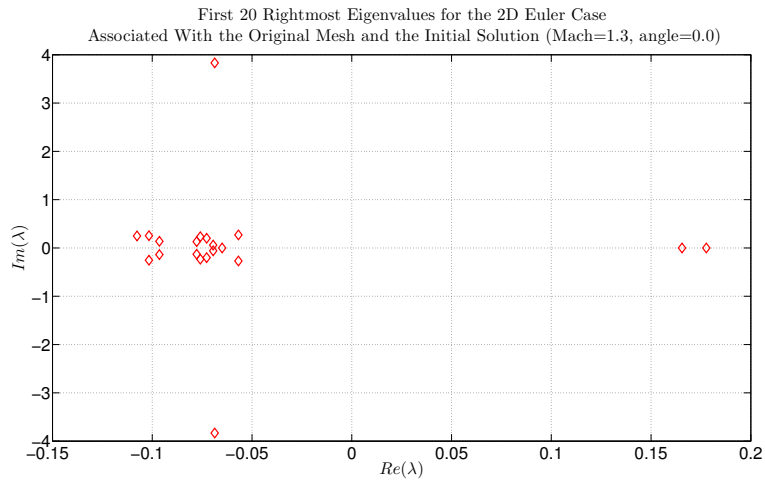


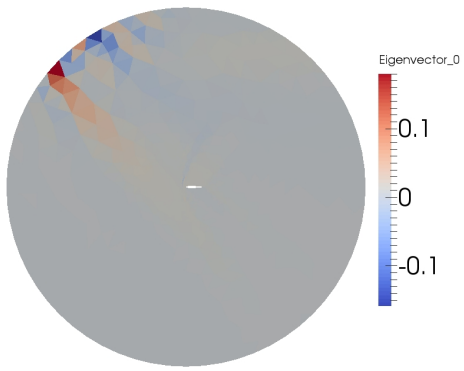
Figure 5.8: First twenty rightmost eigenvalues of the Jacobian matrix of the fourth order scheme at the initial solution for the 2D Euler case with Mach=1.3, angle=0.0 free stream

exists a derivative $\frac{d(\lambda_i)}{d\varepsilon_s}$ larger than 0.0001 and smaller than 2 for the first four eigenvalues, i.e. $i = 0, 1, 2, 3$. The other settings for the optimization problem are as same as those for the 2D Euler case with Mach=1.2, angle=0.0 free stream in Chapter 4.

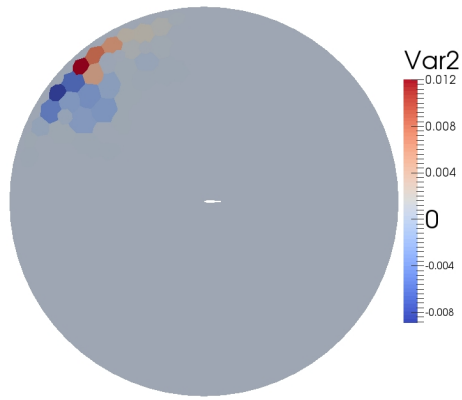
After solving the optimization problem, we obtain the change in x-coordinates shown in Figure (5.10a), and the change in y-coordinates shown in Figure (5.10b). Both the original mesh and the new mesh are shown in Figure (5.11). The new mesh is shown separately in Figure (5.12), and we see the new mesh looks normal. By applying the new mesh to the time-stepping where the initial solution is the solution of the first order scheme, we generate the convergence history shown in Figure (5.13), from which we can see that the original unstable time-stepping has been stabilized. We also plot the eigenvalues of the Jacobian matrix based on the converged solution of the fourth order scheme on the new mesh, shown in Figure (5.14), and compared with the eigenvalues in Figure (4.12), we can see all the unstable eigenvalues have been shifted into the left half plane.

5.3 Summary and Further Discussion

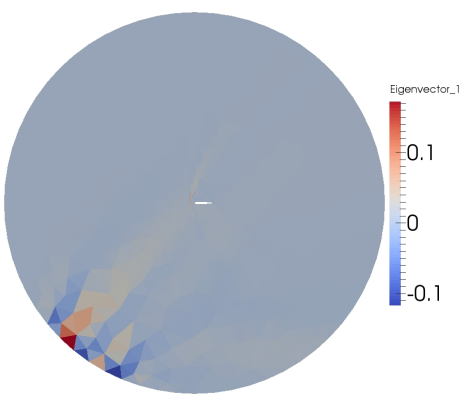
In this chapter, by the same method used for the fixed point stabilization, we successfully stabilized two unstable fourth order cases at the initial converged first or-



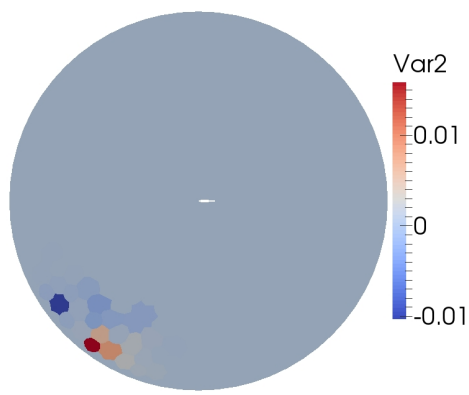
(a) The density component's eigenvector associated with the first rightmost eigenvalue of the Jacobian matrix of the fourth order scheme at the initial solution



(b) The derivatives of the first rightmost eigenvalue's real part with respect to the normalized x-coordinate $\frac{dRe(\lambda_0)}{dx_s}$

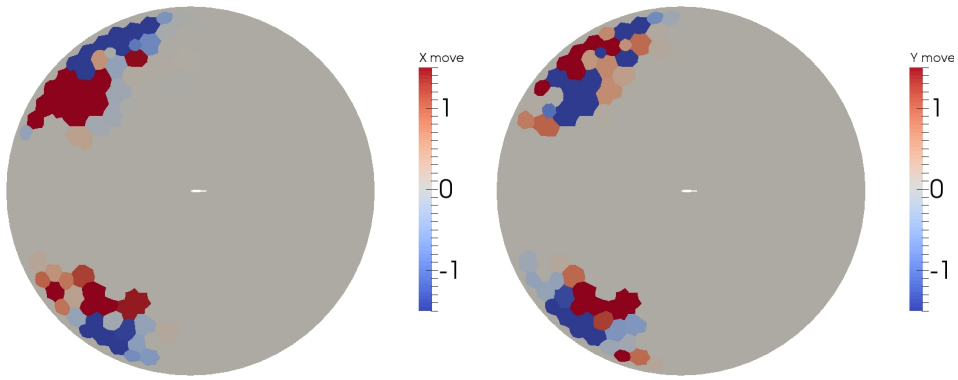


(c) The density component's eigenvector associated with the second rightmost eigenvalue of the Jacobian matrix of the fourth order scheme at the initial solution



(d) The derivatives of the second rightmost eigenvalue's real part with respect to the normalized x-coordinate $\frac{dRe(\lambda_1)}{dx_s}$

Figure 5.9: Eigenvectors and derivatives for the 2D Euler case with Mach=1.3, angle=0.0 free steam



(a) X-coordinate change (b) Y-coordinate change

Figure 5.10: The normalized coordinates's change for non-fixed point stabilization for the 2D Euler case with Mach=1.3, angle=0.0 free stream

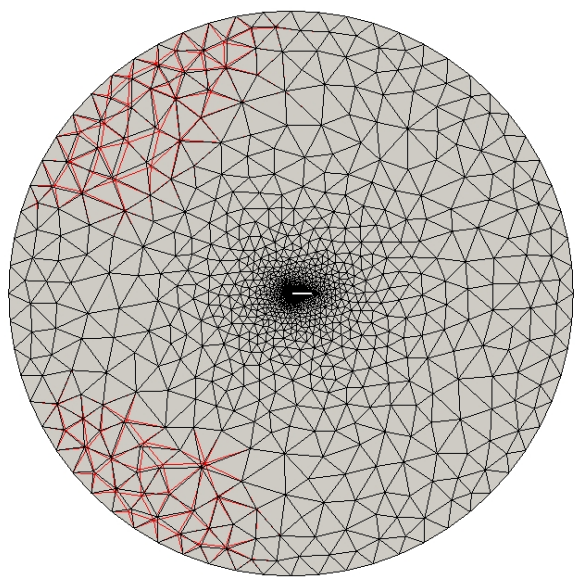


Figure 5.11: The mesh comparison for the non-fixed stabilization for the 2D Euler case with Mach=1.3, angle=0.0 (the red line is for the new mesh, and the black line is for the original mesh)

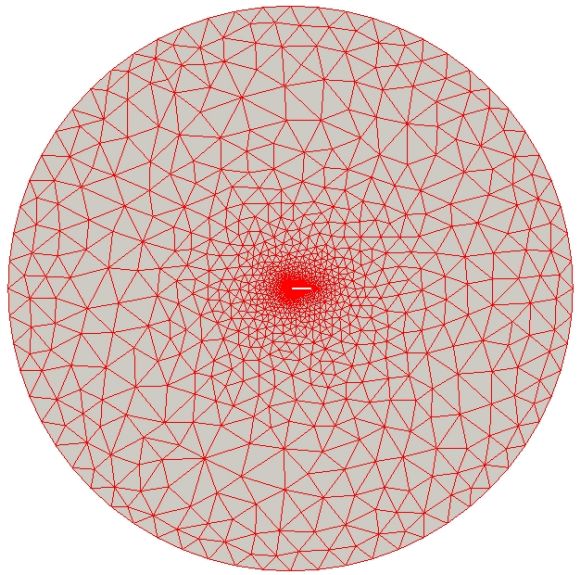


Figure 5.12: The new mesh arising from non-fixed point stabilization for the 2D Euler case with Mach=1.3, angle=0.0 free stream

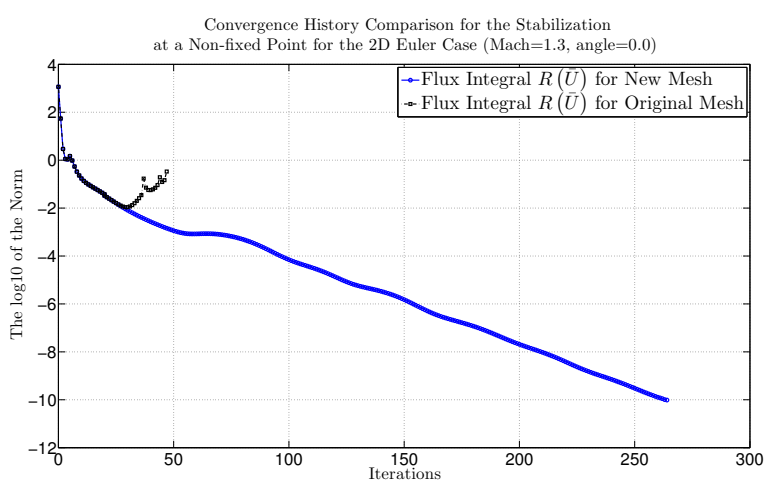


Figure 5.13: Convergence history comparison for the non-fixed stabilization for the 2D Euler case with Mach=1.3, angle=0.0 free stream

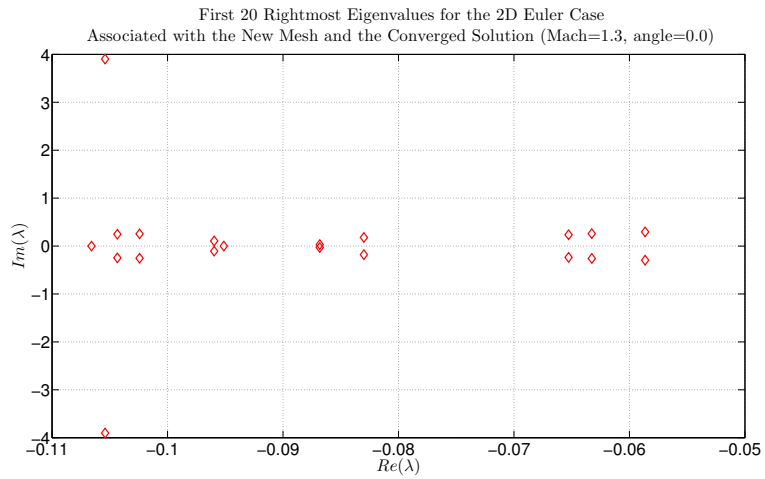


Figure 5.14: First twenty rightmost eigenvalues of the Jacobian matrix on the converged solution of the fourth order scheme on the new mesh for the 2D Euler case with Mach=1.3, angle=0.0 free stream

der solution. In fact, if we compare the results of the non-fixed point stabilization with the results of the fixed point stabilization, we can see they are quite similar. We first compare eigenvalues and eigenvectors. Comparing Figure (5.1) with Figure (3.36), we can see that the first several rightmost eigenvalues of the Jacobian matrix of the fourth order scheme calculated on the converged solution of the fourth order scheme are close to those of the Jacobian matrix of the fourth order scheme calculated on the initial solution. Seen from Figure (4.5), Figure (4.18b), Figure (5.2a), Figure (5.2c), for the eigenvectors associated with the first two rightmost eigenvalues, we can draw similar conclusions as those for the eigenvalues. For the 2D Euler case with Mach=1.3, we can draw similar conclusions for the rightmost eigenvalues and the eigenvectors associated with the first two rightmost eigenvalues as well. The Jacobian matrix depends on the solution. The difference between the converged solution of the fourth order scheme and the converged solution of the first order scheme is small. Therefore, it is reasonable that the eigenvalues and the eigenvectors are similar. For both cases, we can also see similarities for the derivatives of the eigenvalues, the mesh movement, and the first several rightmost eigenvalues after mesh movement.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis, we developed a fixed point stability analysis model and verified its validity.

1. If the largest eigenvalue of the mapping relating the computational error of two successive iterations is real, the rate of convergence of the norm of computational error converges to the norm of the largest eigenvalue. The direction of the computational error converges to the direction of the eigenvector associated with the largest eigenvalue, or the opposite.

2. If the largest eigenvalues of the mapping are a complex conjugate pair, the rate of convergence of the norm of the computational error oscillates around the norm of the largest eigenvalues. The asymptotic decrease rate of the norm of the computational error is consistent with the norm of the largest eigenvalues.

For the flux integral and the solution update, the conclusions are the same. This stability model can be applied to both for linear problems and nonlinear problems.

We also developed a methodology to change the eigenvalues of a matrix in a quantitative and controllable way. Specifically, for the Jacobian matrix arising from the spatial discretization of the Euler equations on an unstructured mesh, we demonstrated that if a coordinate's change is not greater than one-tenth the shortest incident edge length, the changes of the eigenvalues can be predicted within acceptable tolerance. In practice, we may need a larger movement for stabilization.

We used this method to change the eigenvalues of the Jacobian matrix by changing the coordinates of vertices of a mesh. By solving a defined optimization problem, we obtained the movement for the mesh and applied the movement to the time-stepping. The results showed that the unstable cases were successfully stabilized and the unstable eigenvalues are shifted into the stable region. We implemented this stabilization both at the fixed point and a non-fixed point for two 2D Euler cases.

6.2 Future Work for Stability and Stabilization

There are several problems which are mentioned in previous chapters but are not resolved. They are interesting topics for future work.

1. We need to study the reason of the emergence of the singularity of the eigenvalue's derivatives with respect to the coordinates, and the way to deal with it.

2. Only a small part of coordinates are needed for stabilization. We could reduce the time for stabilization significantly if only the coordinates most responsible for the instability are considered.

3. We need to find out why eigenvalue's changes after mesh movement do not coincide with the changes predicted from the optimization problem. To have the eigenvalue's changes predictable, we may need multiple smaller movements for stabilization.

4. For practical application, we may need to blend the stability analysis and the stabilization method with a technique which guarantees the solution is physically valid, for instance, a limiter. Similarly, we can blend the stability analysis and stabilization method with other techniques as well.

It is worth extending the study in this thesis to the 2D Navier-Stokes equations, the 3D Euler equations, the 3D Navier-Stokes equations, and other reconstruction schemes, etc.

It is well known that the matrix method is misleading for ill-conditioned operators and the computation of the eigenvalues of an ill-conditioned matrix is not reliable in finite precision arithmetic. In future work, we may also need to study how to implement stability and stabilization for ill-conditioned operators.

Qualitatively speaking, stability is determined by three aspects: the physical problem, the reconstruction method, and the mesh. It would be useful to develop guidelines for generating a mesh that produces a stable system for a given physical problem and a given reconstruction method.

It would be promising that we improve the stability of a shock wave capturing scheme such that a weaker limiter is sufficient, which is good for resolution of the solution.

6.3 Open Questions: Spectral Analysis and Optimization — Beyond Stability and Stabilization

In general, eigenvalues and eigenvectors have a wide application. It would be natural that the optimization for eigenvalues and eigenvectors will bring wide applications as well. The approach used in this thesis propose a framework to optimize the eigenvalues of a matrix. We can implement similar optimization for other purposes and for other problems as well.

Bibliography

[1] Fred Brauer and John A. Nohel. *The Qualitative Theory of Ordinary Differential Equations: An Introduction*. W. A. Benjamin, Inc., 1969.

[2] Carmen Campos, Jose E. Roman, Eloy Romero, and Andres Tomas. SLEPc homepage. `http://http://www.grycap.upv.es/slepc/`, 2011.

[3] J.G. Charney, R. Fjortoft, and J. Von Neumann. Numerical integration of the barotropic vorticity equation. *Tellus*, 2:237–254, 1950.

[4] J. Crank and P. Nicolson. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Mathematical Proceedings of the Cambridge Philosophical Society*, 43(01):50–67, January 1947.

[5] M. B. Giles. Energy stability analysis of multi-step methods on unstructured meshes. CFDL Report 1, MIT Dept. of Aero. and Astro, 1987.

[6] M. B. Giles. Stability analysis of a Galerkin/Runge-Kutta Navier-Stokes discretisation on unstructured tetrahedral grid. *J. Comput. Phys.*, 132(2):201–214, April 1997.

[7] Florian Haider, Jean-Pierre Croiselle, and Bernard Courbet. Stability analysis of the cell centered finite-volume MUSCL method on unstructured grids. *Numerische Mathematik*, 113(4):555–600, 2009.

[8] Christopher Hill. Personal communication. October 2014.

[9] Charles Hirsch. *Numerical Computation of Internal and External Flows: Fundamentals of Computational Fluid Dynamics*, volume 1. Butterworth-Heinemann, second edition, 2007.

[10] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

[11] P. Lancaster. On eigenvalues of matrices dependent on a parameter. *Numerische Mathematik*, 6(1):377–387, December 1964.

[12] P. D. Lax and R. D. Richtmyer. Survery of the stability of linear finite difference equations. *Commun. Pure Appl. Math.*, 9:267–293, 1956.

[13] H. W. J. Lenferink and M. N. Spijker. On the use of stability regions in the numerical analysis of initial value problems. *Mathematics of Computation*, 57(195):221–237, Jul. 1991.

[14] Doron Levy and Eitan Tadmor. From semidiscrete to fully discrete: stability of Runge-Kutta schemes by the energy method. *SIAM REV.*, 40(1):40–73, 1998.

[15] Mathworks. quadprog documentation. `http://www.mathworks.com/help/optim/ug/quadprog.html`, December 2015.

[16] The MathWorks, Inc., Natick, Massachusetts. *MATLAB version 7.14.0.739 (R2012a)*, 2012.

[17] Christopher Michalak and Carl Ollivier-Gooch. Globalized matrix-explicit Newton-GMRES for the high-order accurate solution of the Euler equations. *Computers and Fluids*, 39:1156–1167, 2010.

[18] P. Moinier and M. B. Giles. Stability analysis of preconditioned approximations of the Euler equations on unstructured meshes. *J. Comput. Phys.*, 178(2), 2002.

[19] Amir Nejat and Carl Ollivier-Gooch. A high-order accurate unstructured finite volume Newton-Krylov algorithm for inviscid compressible flows. *Journal of Computational Physics*, 227(4):2592–2609, 2008.

[20] Carl F. Ollivier-Gooch. GRUMMP — Generation and Refinement of Unstructured, Mixed-element Meshes in Parallel. http://tetra.mech.ubc.ca/GRUMMP, 1998–2010.

[21] Carl F. Ollivier-Gooch. A toolkit for numerical simulation of PDE's: I. Fundamentals of generic finite-volume simulation. *Computer Methods in Applied Mechanics and Engineering*, 192:1147–1175, February 2003.

[22] Satish C. Reddy and Lloyd N. Trefethen. Stability of the method of lines. *Numerische Mathematik*, 62:235–267, 1992.

[23] Robert D. Richtmyer and K. W. Morton. *Difference Method for Initial-value Problems*. Interscience Publishers, second edition, 1967.

[24] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[25] Barry Smith, Satish Balay, Jed Brown, Lisandro Dalcin, Dmitry Karpeev, Matthew Knepley, and Hong Zhang. PETSc home page. `http://www.mcs.anl.gov/petsc`, 2011.

[26] Bram van Leer. Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection. *J. Comput. Phys.*, 23(3):276–299, March 1977.

[27] Bram van Leer. Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method. *Journal of Computational Physics*, 32:101–136, 1979.

Appendix A

The Mappings for the Solution Update and the Flux Integral

A.1 The Mapping for the Solution Update

Recalling Equation (2.26), we have

$$\left(\frac{I}{\Delta t^{k+1}} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k+1} = -\left(\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\left(\Delta\bar{U}^k\right)\right) + O\left(\Delta\bar{U}^k\right)^2.$$

Replace k by $k-1$ to get

$$\left(\frac{I}{\Delta t^{k}} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k} = -\left(\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\left(\Delta\bar{U}^{k-1}\right)\right) + O\left(\Delta\bar{U}^{k-1}\right)^2.$$

Combining these two equations and dropping the high order terms yield

$$\begin{aligned}&\left(\frac{I}{\Delta t^{k+1}} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k+1} - \left(\frac{I}{\Delta t^{k}} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k}\\ &= -\left(\left(\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\left(\Delta\bar{U}^{k}\right)\right) - \left(\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\left(\Delta\bar{U}^{k-1}\right)\right)\right).\end{aligned} \tag{A.1}$$

Retrieve Equation (2.12) and Equation (2.22)

$$\bar{U}^k = \bar{U}^{k-1} + \alpha\delta\bar{U}^k,$$

$$\bar{U}_p^k = \bar{U}_s + \Delta\bar{U}^k.$$

Combine these two equations to obtain

$$\Delta\bar{U}^k - \Delta\bar{U}^{k-1} = \alpha\delta\bar{U}^k. \tag{A.2}$$

Using Equation (A.2) in Equation (A.1), we have

$$\left(\frac{I}{\Delta t^{k+1}}+\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k+1}=\left(\frac{I}{\Delta t^{k}}+\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k}-\alpha\left(\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\left(\delta\bar{U}^{k}\right)\right).$$

A compound form is

$$\left(\frac{I}{\Delta t^{k+1}}+\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k+1}=\left(\frac{I}{\Delta t^{k}}+(1-\alpha)\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k}. \tag{A.3}$$

For backward Euler time-stepping, Δt is constant, and by replacing k by $k-1$, we have

$$\left(\frac{I}{\Delta t}+\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k}=\left(\frac{I}{\Delta t}+(1-\alpha)\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\delta\bar{U}^{k-1}. \tag{A.4}$$

A.2 The Mapping for the Flux integral

The computational solution at the kth iteration is denoted by $\bar{U}^k$, and based on this solution, the flux integral is

$$R\left(\bar{U}^k\right). \tag{A.5}$$

Recall Equation (2.22)

$$\bar{U}_p^k=\bar{U}_s+\Delta\bar{U}^k.$$

Use Equation (2.22) and apply the Taylor explanation to get

$$R\left(\bar{U}^k\right)=R\left(\bar{U}_s+\Delta\bar{U}^k\right)=R\left(\bar{U}_s\right)+\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|\left(\Delta\bar{U}^k\right).$$

Since $R\left(\bar{U}_s\right)=0$, we have

$$R\left(\bar{U}^k\right)=\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\left(\Delta\bar{U}^k\right). \tag{A.6}$$

Replacing k by $k-1$, we have

$$R\left(\bar{U}^{k-1}\right)=\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\left(\Delta\bar{U}^{k-1}\right). \tag{A.7}$$

Recall Equation (2.31)

$$\left(\frac{I}{\Delta t^k} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\Delta U^k = \left(\frac{I}{\Delta t^k} + (1-\alpha)\left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)\Delta \bar{U}^{k-1}.$$

Apply Equation (A.6) and Equation (A.7) to obtain

$$\left(\frac{I}{\Delta t^k}\right)\left(\Delta U^k - \Delta \bar{U}^{k-1}\right) = (1-\alpha)R\left(\bar{U}^{k-1}\right) - R\left(\bar{U}^k\right). \tag{A.8}$$

By using equation (A.2) in equation (A.8), we have

$$\left(\frac{I}{\Delta t^k}\right)\left(\alpha \delta \bar{U}^k\right) = (1-\alpha)R\left(\bar{U}^{k-1}\right) - R\left(\bar{U}^k\right). \tag{A.9}$$

In this thesis, $\alpha = 1$ and Δt is constant; so we have

$$\left(\frac{I}{\Delta t^k}\right)\left(\delta \bar{U}^k\right) = -R\left(\bar{U}^k\right), \tag{A.10}$$

or

$$\left(\delta \bar{U}^k\right) = -\Delta t R\left(\bar{U}^k\right). \tag{A.11}$$

Substituting Equation (A.11) into Equation (A.4) and applying $\alpha = 1$, we have

$$\left(\frac{I}{\Delta t} + \left.\frac{\partial \bar{R}}{\partial \bar{U}}\right|_{\bar{U}=\bar{U}_s}\right)R\left(\bar{U}^k\right) = \left(\frac{1}{\Delta t}\right)R\left(\bar{U}^{k-1}\right). \tag{A.12}$$

Appendix B

Angle Error Estimation

The angle between the computational error and the eigenvector x_0 is defined as

$$\theta = \arccos\left(\frac{x_0^* \Delta\bar{U}^k}{\left|x_0^*\right| \left|\Delta\bar{U}^k\right|}\right). \tag{B.1}$$

Applying the decomposition for $\Delta\bar{U}^k$ by Equation (2.48), for $x^*\Delta\bar{U}^k$, we have

$$\begin{aligned} x_0^*\Delta\bar{U}^k &= x_0^*\left(r_0^k \sum_{i=0}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right) = x_0^* r_0^k \left(x_0 + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right) \\ &= r_0^k \left(x_0^* x_0 + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_0^* x_i\right). \end{aligned} \tag{B.2}$$

Similarly, for $\left|x_0^*\right| \left|\Delta\bar{U}^k\right|$, we have

$$\left|x_0^*\right| \left|\Delta\bar{U}^k\right| = \left|x_0^*\right| \left| r_0^k \sum_{i=0}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i \right|. \tag{B.3}$$

Implementing further deduction, we have

$$\left|x_0^*\right| \left| r_0^k \sum_{i=0}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i \right| = \sqrt{\left(x_0^* x_0\right)} \sqrt{\left(\left(r_0^k \left(\sum_{i=0}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right)\right)^* r_0^k \left(\sum_{i=0}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right)\right)}. \tag{B.4}$$

Therefore, we have

$$\begin{aligned} \left(\left|x_0^*\right| \left| r_0^k \sum_{i=0}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i \right|\right)^2 &= x_0^* x_0 \left(r_0^k \left(\sum_{i=0}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right)\right)^* r_0^k \left(\sum_{i=0}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right) \\ &= r_0^{2k} \left(x_0^* x_0\right) \left(x_0 + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right)^* \left(x_0 + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right) \end{aligned}$$

$$= r_0^{2k}\left(x_0^* x_0\right)\left(x_0^* x_0 + 2\sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i^* x_0 + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^{2k} e^{2Ik\theta_i} x_i^* x_i + O\left(\frac{r_1}{r_0}\frac{r_2}{r_0}\right)^k\right)$$

$$= r_0^{2k}\left(\left(x_0^* x_0\right)\left(x_0^* x_0\right) + 2\sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i}\left(x_i^* x_0\right)\left(x_0^* x_0\right) + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^{2k} e^{2Ik\theta_i}\left(x_i^* x_i\right)\left(x_0^* x_0\right) + O\left(\frac{r_1}{r_0}\frac{r_2}{r_0}\right)^k\right)$$

$$\begin{aligned} = \quad & r_0^{2k}\left(\left(x_0^* x_0\right)\left(x_0^* x_0\right) + 2\sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i}\left(x_i^* x_0\right)\left(x_0^* x_0\right)\right. \\ & \left. + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^{2k} e^{2Ik\theta_i}\left(x_i^* x_0\right)\left(x_i^* x_0\right) + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^{2k} e^{2Ik\theta_i}\left(\left(x_i^* x_i\right)\left(x_0^* x_0\right) - \left(x_i^* x_0\right)\left(x_i^* x_0\right)\right) + O\left(\frac{r_1}{r_0}\frac{r_2}{r_0}\right)^k\right) \end{aligned}$$

$$= r_0^{2k}\left(\left(\left(x_0^* x_0\right) + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i}\left(x_i^* x_0\right)\right)^2 + O\left(\frac{r_1}{r_0}\right)^{2k}\right). \tag{B.5}$$

As a result, for $|x_0^*|\left|r_0^k \sum_{i=0}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right|$, we have

$$|x_0^*|\left|r_0^k \sum_{i=0}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_i\right| = r_0^k\left(\left|\left(x_0^* x_0\right) + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i}\left(x_i^* x_0\right)\right| + O\left(\frac{r_1}{r_0}\right)^{2k}\right). \tag{B.6}$$

Therefore, we have

$$\frac{x_0^* \Delta\bar{U}^k}{\left|x_0^*\right|\left|\Delta\bar{U}^k\right|} = \frac{r_0^k\left(x_0^* x_0 + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_0^* x_i\right)}{r_0^k\left(\left|\left(x_0^* x_0 + \sum_{i=1}^{i=n-1} a_i^0 \left(\frac{r_i}{r_0}\right)^k e^{Ik\theta_i} x_0^* x_i\right)\right| + O\left(\frac{r_1}{r_0}\right)^{2k}\right)} = \pm 1 + O\left(\left(\frac{r_1}{r_0}\right)^{2k}\right). \tag{B.7}$$

For the angle, we have

$$\theta = \arccos\left(\frac{x_0^* \Delta\bar{U}^k}{\left|x_0^*\right|\left|\Delta\bar{U}^k\right|}\right) = \arccos\left(\pm 1 + O\left(\left(\frac{r_1}{r_0}\right)^{2k}\right)\right) = \pi \text{ or } 0 + O\left(\left(\frac{r_1}{r_0}\right)^k\right). \tag{B.8}$$