Error Estimation and Mesh Adaptation Paradigm for Unstructured Mesh Finite Volume Methods

by

Mahkame Sharbatdar

B.Sc., Mechanical Engineering, University of Tehran, 2010
M.ASc., Mechanical Engineering, The University of British Columbia, 2012

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

DOCTOR OF PHILOSOPHY

in

The Faculty of Graduate and Postdoctoral Studies

(Mechanical Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

January 2017

Abstract

Error quantification for industrial CFD requires a new paradigm in which a robust flow solver with error quantification capabilities reliably produces solutions with known error bounds. Error quantification hinges on the ability to accurately estimate and efficiently exploit the local truncation error. The goal of this thesis is to develop a reliable truncation error estimator for finite-volume schemes and to use this truncation error estimate to improve flow solutions through defect correction, to correct the output functional, and to adapt the mesh.

We use a higher-order flux integral based on lower order solution as an estimation of the truncation error which includes the leading term in the truncation error. Our results show that using this original truncation error estimate is dominated by rough modes and fails to provide the desired convergence for the applications of defect correction, output error estimation and mesh adaptation. So, we tried to obtain an estimate of the truncation error based on the continuous interpolated solution to improve their performance. Two different methods for interpolating were proposed: CGM's 3D surfaces and C^1 interpolation of the solution.

We compared the effectiveness of these two interpolating schemes for defect correction and using C^1 interpolation of the solution for interpolating is more helpful compared to CGM, so we continued using C^1 interpolation for other purposes. For defect correction, although using the modified truncation error does not improve the order of accuracy, significant quantitative improvements are obtained.

Output functional correction is based on the truncation error and the adjoint solution. Both discrete and continuous adjoint solutions can be used for functional correction. Our results for a variety of governing equations suggest that the interpolating scheme can improve the correction process significantly and improve accuracy asymptotically.

Different adaptation indicators were considered for mesh adaptation and our results show that the estimate of the truncation error based on the interpolated solution is a more accurate indicator compared to the original truncation error. Adjoint-based mesh

adaptation combined with modified truncation error provides even faster convergence of the output functional.

Preface

The research ideas and methods explored in this thesis are the fruits of a close working relationship between Dr. Carl Ollivier-Gooch and Mahkame Sharbatdar. The implementation of the methods, the data analysis, and the manuscript preparation were done by Mahkame Sharbatdar with invaluable guidance from Carl Ollivier-Gooch throughout the process. The following papers, directly related to this PhD thesis, have been published or are in the submission process.

- M. Sharbatdar, A. Jalali, C. Ollivier-Gooch, "Smoothed Truncation Error in Functional Error Estimation and Correction using Adjoint Methods in an Unstructured Finite Volume Solver". Computers & Fluids, 140:406-421, 2016. Some of the results from this paper are included in Chapter 5.
  A. Jalali is a PhD student in our research group who developed the discrete adjoint solver for his PhD thesis. I developed the continuous adjoint solver, and truncation error estimate used in this work, run all the cases, and wrote the paper.

- M. Sharbatdar, C. Ollivier-Gooch, "Adjoint-based Functional Correction for Unstructured Mesh Finite Volume Methods". submitted, 2016. Some of the results from this paper are included in Chapters 5 and 6.

- M. Sharbatdar, C. Ollivier-Gooch, "Mesh Adaptation Using Smoothed Truncation Error for Unstructured Mesh Finite Volume Methods". In preparation for submission. Some of the results from this paper are included in Chapter 6.

The following paper uses the eigenanalysis of the Jacobian, described in Chapter 2, as a tool to relate the two forms of the numerical error. This paper is the outcome of A. Jalali's PhD thesis.

- A, Jalali, M. Sharbatdar, C. Ollivier-Gooch, "Accuracy Analysis of Unstructured Finite Volume Discretization Schemes for Diffusive Fluxes", Computers & Fluids, 101:220-232, 2014.

Table of Contents

Appendices

List of Tables

List of Figures

Nomenclature

Roman Symbols

$\mathbf{x}$	Right eigenvector
$\mathbf{y}$	Left eigenvector
A	Convective flux Jacobian matrix
A	area
a	Weights in eigendecomposition of truncation error
B	Boundary operator
b	Weights in eigendecomposition of discretization error
C	Boundary weight operator
c	Argyris element coeefficients
D	Viscous flux Jacobian matrix
f	Primal source term
g	Adjoint source term
h	Enthalpy
I	Identity matrix
J	Output functional
L	Linear operator
l	Length scale
Ma	Mach number

P	Pressure
Pr	Prandtl number
R	Reconstruction matrix
r	position vector
Re	Reynolds number
s	length of control volume edges
T	Temperature
t	time
u, v	velocity components
x, y	Cartesian coordinates
Z	Adjoint solution
$\vec{F}$	flux vector
$\hat{n}$	unit normal vector
$\vec{U}$	solution vector

Greek Symbols

α	Angle of attack
γ	Specific heat ratio
λ	Eigenvalues, Taylor series expansion coefficient
μ	viscosity
ρ	Density
τ	Truncation error
ε	Discretization error
φ	Basis functions

ξ, η Coordinates in the reference element

Superscripts

L Left side

n non-smoothed truncation error

R Right side, reconstructed function

s smoothed truncation error

Subscripts

c Correction based on continuous adjoint solution

CS control surface

CV control volume

d Correction based on discrte adjoint solution

h Mesh characteristic size

p Order of accuracy, error correspondong to the primal problem

Acknowledgments

First I would like to thank my supervisor, Dr. Carl Ollivier-Gooch for his constant availability for discussion and for his invaluable help throughout the course of this research.

I would also like to thank my lovely family for their endless encouragement and love: the kindest person I have ever known, my mother, Mahtab; the most reliable person I have ever talked to, my father, Amir; and the dearest brother in all the world, Kamran.

Finally, my deepest appreciation is for my husband, Alireza, for his limitless patience, support, encouragement and love in my life.

Chapter 1

Introduction

Fluid dynamics is described by a set of Partial Differential Equations (PDEs) and as a result of their complexity, there is not any general analytical solution for them. Therefore, the flow properties need to be predicted by physical experiment or numerical solution of the flow equations. Since for most real applied cases, the solution has a multi-scale nature, the cost of experimental analysis is too much. Considering these limitations, numerical solutions are now commonly used as an important analysis tool in the solution of scientific and engineering problems. While the efficiency of such methods has been dramatically improved, there is still a strong industrial desire to produce more accurate numerical solutions.

Computational Fluid Dynamics (CFD) has shown remarkable capability for fluid analysis and also produces accurate and efficient results, both in terms of memory usage and time. Nowadays, CFD is used extensively and successfully in industry throughout the design process, from preliminary design to shape optimization. Increasing computing power has resulted in the development of new computational techniques and algorithms, enhancing the versatility of CFD applications.

Modeling, mesh generation and numerical solution are the three essential components involved in numerical simulations:

- Modeling: The partial differential equations describing the physics of the problem must be defined. Modeling includes these governing equations, the problem geometry, the initial condition and the boundary conditions.

- Mesh Generation: To solve the PDEs numerically, the continuous domain being studied should be tessellated into shapes recognized by the solver. This process, referred to as mesh generation, is one of the most time consuming parts of CFD solution. Since the discretization of the fluid flow equations is carried out on the mesh, the CFD solution will be inaccurate without a proper domain discretization.

- Numerical Solution: Once the PDEs describing the problem are defined and discretized over the domain in the form of a large system of algebraic equations, the

system of equations is solved and the numerical solution for the flow of interest is obtained.

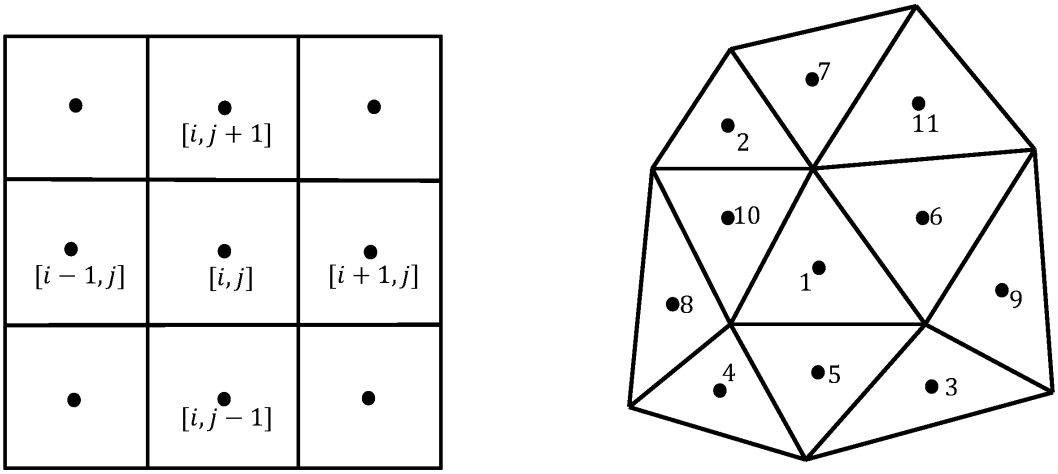
Greater emphasis is being placed on quantifying the accuracy of numerical solutions as the complexity of problems simulated using CFD has grown. The two error terms describing the accuracy of the scheme are discretization error and truncation error and these error terms are the main focus of this thesis. Historically, the main tool for quantifying solution accuracy has been grid refinement studies. However, for real-world application problems, the baseline simulation often pushes the limits of available computing resources, so the cost of finer mesh simulation for a grid refinement study is prohibitive. Such studies are also complicated by modeling issues, notably in the near wall behavior of turbulence models. Meeting this need for error quantification for industrial scale CFD requires a new paradigm in which a robust flow solver with adaptive and error quantification capabilities reliably produces solutions with known error bounds with minimal human intervention.

The goal of this thesis is to prototype methods for error quantification in a framework consistent with numerics employed in modern commercial CFD solvers. Commercial CFD software typically handles complex geometry using unstructured meshes (Section 1.1), for which accuracy issues are not as well understood as for structured meshes. Furthermore, commercial CFD software currently uses finite volume method almost exclusively; the essential parts of the finite volume method is described in Section 1.2. The error quantification paradigm discussed in this thesis is based on the truncation error. The definition of the truncation error and how it is different form the discretization error is described in Section 1.3. Section 1.4 is devoted to different techniques for estimating the truncation error for use in improving flow solutions through defect correction, estimating the solution functional error and driving the mesh adaptation as discussed in Section 1.5. Finally, this chapter concludes by describing the objective of the thesis in details, Section 1.6, and providing an overview of the thesis, Section 1.7.

1.1 Structured and Unstructured Meshes

In a structured mesh, all cells and vertices have the same fixed and predictable topology. As a result, each cell and vertex can be identified by indices. As an example, in a structured mesh, Figure 1.1a, cell (i, j) is always topologically between cell $(i + 1, j)$ and cell $(i - 1, j)$ which are located on the topological right and left of the reference

cell, respectively.



(a) Structured mesh

(b) Unstructured mesh

Figure 1.1: Mesh topology

Unstructured meshes on the other hand have elements with irregular and variable topologies in the sense that the connectivity is not predictable and must be explicitly stored. For an unstructured mesh, Figure 1.1b, there is no particular topology which indicates the neighbors of control volume 1, for instance. The mesh adjacency data must therefore be stored to know the neighbors of a given cell.

From the numerical point of view, structured meshes are attractive as a result of their predictable topology, implying that storing the neighboring cells is not necessary. Besides, as a result of the regular topology, the linear system arising from implicit time discretization have a simple matrix structure and can be solved faster. Furthermore, a fixed interpolation, or extrapolation, can be used for flux integration. For more clarification, flux computation in finite volume method is considered: a fixed template is sufficient for flux computation in structured meshes; accordingly, faster solution with reduced memory usage can be developed by capitalizing on their predictable topology [27]. However, the task of generating structured meshes around complex configurations has proved to be a considerable challenge. Obtaining a structured mesh of a complex domain often involves time-consuming human intervention that can easily offset the saving realized by the solver itself. Furthermore, generalizing mesh adaptation in this approach is still not an easy task and needs significant effort.

On the other hand, for unstructured meshes evaluating the same flux as discussed above for finite volume method requires performing a polynomial reconstruction of the solution over each control volume. In spite of the higher cost for the solver, unstructured meshes are generally much easier to generate automatically around arbitrary complex geometries. Moreover, unstructured meshes are more flexible than structured meshes in refinement based on the geometry and adaptation based on the solution features and gradients. Accordingly, the time and effort required by the user to produce an acceptable mesh can be reduced significantly [51]. These features of structured and unstructured meshes are summarized in Table 1.1.

	Structured mesh	Unstructured mesh
Memory usage	less memory	more memory
Stencil	fixed	variable
Topology	regular	irregular
Solution time	shorter	longer
Mesh generation	significant effort	less effort
Adaptability	significant effort	less effort
Industrial usage trend	less common	more common

Table 1.1: Comparison of features of structured and unstructured meshes

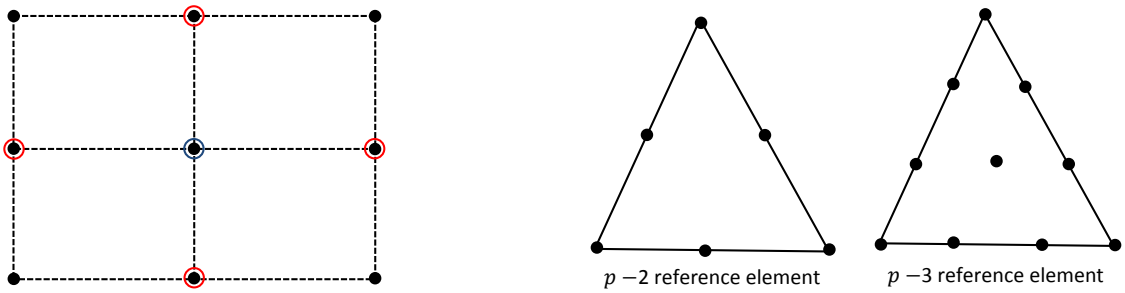
Considering all these differences, unstructured meshes are the most common choice for complex geometries; associated solvers are becoming more common in modern CFD applications and promise to be more capable and successful for complex aerodynamic problems.

1.2 The Finite Volume Method

Once the flow field is discretized and the mesh, either structured or unstructured, is generated, the fluid flow equations should be solved over the discretized domain. The fluid flow equations are generally discretized by finite difference, finite element or finite volume methods.

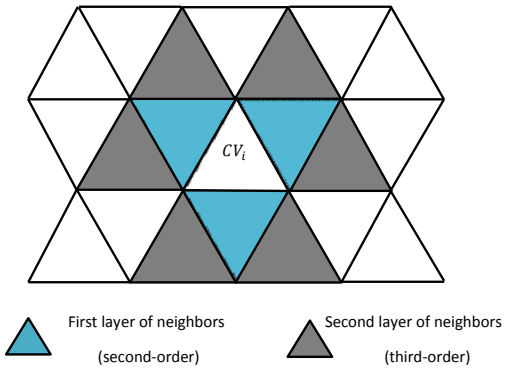
The finite difference method uses a point-wise representation of the flow field. The flow equations are solved only for variables defined at mesh vertices and the point-wise solution derivatives are approximated using a difference relation obtained by the Taylor series expansion of the solution. Figure 1.2a depicts a five-point stencil that might be used in a 2D centered finite difference method. The finite difference scheme

was the original approach to the CFD problems and is well suited for structured grids. However, it is difficult to be implemented on unstructured meshes and moreover, it does not necessarily conserve mass, momentum, and energy of the flow, implying that it is not a proper choice for some practical applications such as flows with shocks.



(a) Finite difference method

(b) Finite element method



(c) Finite volume method

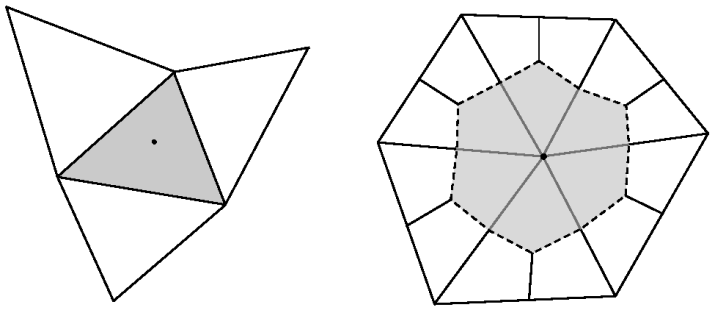
Figure 1.2: Discretization methods

The finite element method is a well established mathematical approach for numerical solution of PDEs. In this method, equations are multiplied by a test function and then integrated over the discretized domain. The finite element solution is represented by local basis functions over the elements of the mesh. The extension of the method to higher-order solutions is as easy as using higher-order degree test and basis functions, Figure 1.2b. As implementing the approach to arbitrary shapes is not an issue, it can be extended to unstructured meshes as well. Similar to the finite difference method, the challenge with using continuous Galerkin methods for the finite element approach is conservation of mass, momentum and energy; although conservation is achievable, it

is not an easy task particularly in flows with discontinuities.

The finite volume approach is based on control volume analysis and discretizes the governing equations in integral form. Consequently, it conserves mass, momentum and energy. As a result of the conservation property, the capability of the finite volume method in capturing discontinuities in the flow is enhanced. The solution is represented by reconstructed piecewise polynomials based on the control volume averages and the governing equations are discretized the by computing a discrete flux integral. Similar to the finite element method, it is very flexible for complex geometries and unstructured meshes. Higher-order application of finite volume methods is possible by using a higher-order polynomial inside each control volume [57], Figure 1.2c. However, high-order finite volume methods are more complicated compared to finite element method; we will discuss this in more detail later.

Finite volume solvers can support two common types of control volumes that are widely used to tessellate an unstructured mesh. Cell-centered control volumes are the same as geometric cells of the mesh and the geometric cell centroid is typically chosen as the reference location of the control volume, Figure 1.3a. In vertex-centered control volumes, the mesh vertices are chosen as the control volume reference point and the control volume is defined by connecting cell centroids with face centroids of the neighboring cells in the primal mesh, Figure 1.3b.



(a) Cell-centered control volume (b) Vertex-centered control volume

Figure 1.3: Control Volume illustration

Among the three methods described above for discretizing the governing equation, we are using the finite volume approach as a result of its capability in simulating complex flows and the fact that most commercial CFD software currently uses this method, particularly for compressible flows.

1.2.1 Spatial Discretization

To discretize the flow equations using the finite volume method, the governing equations should be recast in fully conservative form as:

$$\frac{\partial \vec{U}}{\partial t} + \nabla \cdot \vec{F} = 0 \tag{1.1}$$

where $\overrightarrow{U}$ denotes the solution vector and $\overrightarrow{F}$ is the flux vector which is a combination of both convective, $\vec{F}_c$, and diffusive flux, $\vec{F}_v$ vectors.

$$\vec{F} = \vec{F}_c - \vec{F}_v \tag{1.2}$$

Integrating Eq. 1.1 over an arbitrary control volume and using the divergence theorem gives the 2D finite volume formulation of the governing equations in the form of a volume integral and a surface integral

$$\iint_{CV} \frac{\partial \vec{U}}{\partial t} dA + \oint_{CS} \vec{F} \cdot \hat{n}\, ds = 0 \tag{1.3}$$

where $\hat{n}$ represents the outward unit normal vector, CV and CS are the control volume and control surface (the boundary of the control volume), respectively and ds is the infinitesimal. Assuming that the discretized physical domain does not change in time, U can be brought out from the integral in Eq. 1.3, as the average solution vector of the control volume

$$\frac{d\overline{U}_i}{dt} = -\frac{1}{A_{CV_i}} \oint_{CS_i} \vec{F} \cdot \hat{n}\, ds \tag{1.4}$$

The left hand side is the time derivative of the average solution vector in the i^{th} control volume. The right hand side is called the flux integral or residual of the control volume and represents the spatial discretization of the same control volume. The flux integral is a function of the solution vector and can be re-written as

$$\frac{d\overline{U}_i}{dt} = -R_i\left(\overline{U}_i\right) \tag{1.5}$$

1.2.2 Solution Reconstruction

To obtain an accurate numerical approximation to the residual, we reconstruct a polynomial representation of the solution within each control volume. The solution within each control volume is represented by the Taylor series expansion:

$$\begin{aligned} U_i^R(x-x_i, y-y_i) &= U|_i + \left.\frac{\partial U}{\partial x}\right|_i (x-x_i) + \left.\frac{\partial U}{\partial y}\right|_i (y-y_i) + \left.\frac{\partial^2 U}{\partial x^2}\right|_i \frac{(x-x_i)^2}{2} \\ &+ \left.\frac{\partial^2 U}{\partial x \partial y}\right|_i (x-x_i)(y-y_i) + \left.\frac{\partial^2 U}{\partial y^2}\right|_i \frac{(y-y_i)^2}{2} + \ldots \end{aligned} \tag{1.6}$$

where U_i^R is the value of the reconstructed solution and $\frac{\partial^{k+1} U_i}{\partial x^k y^l}$ are its derivatives at the reference point (x_i, y_i) of control volume i. These values are the coefficients of the Taylor polynomials and the degree of the polynomial determines the order of accuracy of the solution. To obtain conservation of the mean within a control volume, we need to satisfy

$$\begin{aligned} \overline{U}_i = \frac{1}{A_i}\int_{V_i} U_i^R dA &= U|_i + \left.\frac{\partial U}{\partial x}\right|_i \overline{x}_i + \left.\frac{\partial U}{\partial y}\right|_i \overline{y}_i + \left.\frac{\partial^2 U}{\partial x^2}\right|_i \frac{\overline{x^2}_i}{2} \\ &+ \left.\frac{\partial^2 U}{\partial x \partial y}\right|_i \overline{xy}_i + \left.\frac{\partial^2 U}{\partial y^2}\right|_i \frac{\overline{y^2}_i}{2} + \ldots \end{aligned} \tag{1.7}$$

where $\overline{x^p y^q}_i$ are the moments of area:

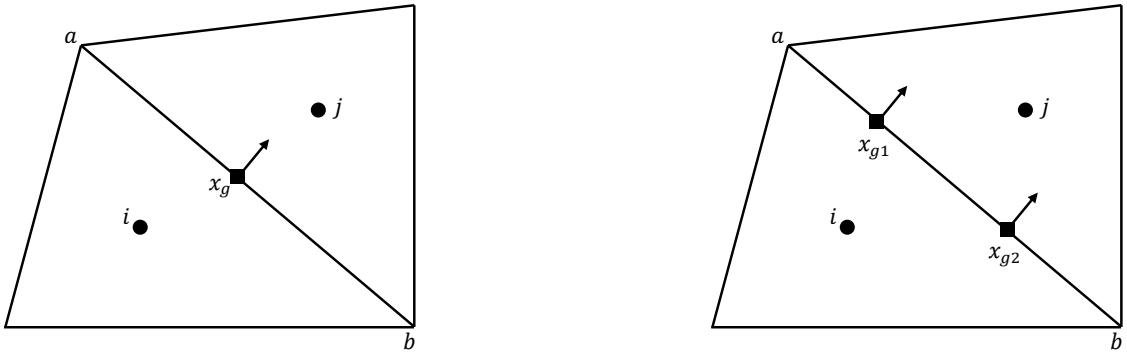
$$\overline{x^p y^q}_i = \frac{1}{A_i}\int_{V_i} (x-x_i)^p (y-y_i)^q dA. \tag{1.8}$$

The error magnitude or the difference between the actual control volume average $\overline{U}_j$ and the average of U_i^R for control volumes in the stencil $\{V_j\}_i$ should be minimized [66]. Eq. 1.7 is written for every control volume within the stencil of control volume i. The number of these control volumes must be more than the number of reconstructed solution coefficients to yield a least-squares system. Once the polynomial coefficients are known, the reconstructed flow properties at any point in the control volume can easily be found using Eq. 1.6. This is a general overview of the solution reconstruction and more details are available in literature [9, 52].

1.2.3 Flux Integration

To compute the flux integral for each control volume, Eq. 1.5, numerical fluxes should be integrated over control volume edges. The accuracy of flux integration should be equal or higher than the accuracy of solution reconstruction. This integration is done by the Gauss quadrature integration rule. Gauss quadrature gives the capability of accurately evaluating a definite integral with the integrand evaluated at only a few points [88].

Consider the two interior control volumes shown in Figure 1.4. The flux vector is calculated by solution reconstruction described in previous section and the numerical flux formula. The conservation property of the finite volume method requires that the flux leaving a control volume must enter its neighbor, so along edge $\overline{ab}$, the flux leaving control volume i should enter control volume j. However, the solutions and gradients reconstructed at the two sides of a Gauss point are not necessarily equal. Therefore, a flux function is essential to yield a unique flux vector. By knowing the solutions and gradients by solution reconstruction and the unit normal by the geometry of the mesh, the flux integral can be evaluated by the numerical flux formula.



(a) Second-order

(b) Third and fourth order

Figure 1.4: Cell-centered flux calculation

For the second order solution or linear reconstruction case, one quadrature point per edge is used which is at the middle of the corresponding edge. The flux integral for each edge can be approximated by knowing the numerical flux at the edge midpoint and the length of the edge. For higher order cases, more quadrature points per edge are required which are used with the proper weightings. Comprehensive information regarding the locations and weights of the Gauss quadrature integration points has been

given by Van Altena [92]. The control volume flux integral in Eq. 1.4 is approximated as the summation of flux integrals over the edges and the flux is the subtracted from the flux integral for control volume i and added to the flux integral for control volume j, following the direction of the normal.

Convective Fluxes

Consider the advective equation that describes the two-dimensional transport of a conserved scalar ϕ in a constant velocity field $\vec{V} = (u, v)$.

$$\frac{\partial \phi}{\partial t} + u\frac{\partial \phi}{\partial x} + v\frac{\partial \phi}{\partial y} = 0 \tag{1.9}$$

The solution and flux vector are

$$U = \phi \quad , \quad \vec{F} = \begin{pmatrix} u\phi \\ \\ v\phi \end{pmatrix} \tag{1.10}$$

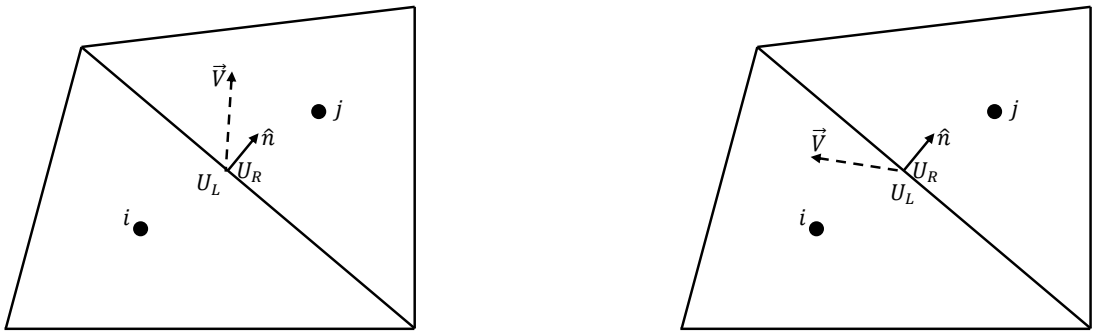
and the flux vector is only dependent on the solution and not the solution gradient. There are two major schemes that are used for numerical convective flux computation: central and upwind schemes.

For the central flux formulation, the flux function is simply the arithmetic average of the two flux vectors computed from each side, $\vec{F}\left(U^L\right)$ and $\vec{F}\left(U^R\right)$, :

$$\vec{F}_c = \frac{\vec{F}\left(U^L\right) + \vec{F}\left(U^R\right)}{2} \tag{1.11}$$

Considering that information in the flow field is propagating at the wave speed, the solution at each point is influenced by the solution upstream and the solution downstream does not physically affect it. The upwind scheme gives the numerical flux at the face quadrature points based on the direction of the velocity vector at those points. For instance, in Figure 1.5, the numerical flux at the common edge is given as:

$$\vec{F}_{ab} = \begin{cases} \vec{F}_L & \vec{V} \cdot \hat{n} \geq 0 \\ \vec{F}_R & \vec{V} \cdot \hat{n} < 0 \end{cases} \tag{1.12}$$



(a) $\vec{V} \cdot \hat{n} > 0 \Rightarrow \vec{F} = \vec{F}_L$ (b) $\vec{V} \cdot \hat{n} < 0 \Rightarrow \vec{F} = \vec{F}_R$

Figure 1.5: Upwind flux illustration

Diffusive Fluxes

Consider the general form of unsteady two-dimensional diffusion

$$\frac{\partial \phi}{\partial t} - \left(\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} \right) = 0. \tag{1.13}$$

Recasting this equation in the form of Eq. 1.1 yields the unknown solution and flux vector as:

$$U = \phi \quad , \quad \vec{F} = \begin{pmatrix} -\frac{\partial \phi}{\partial x} \\ \\ -\frac{\partial \phi}{\partial y} \end{pmatrix} = -\nabla \phi \tag{1.14}$$

and the flux vector is only dependent on gradients. For diffusive fluxes, the face gradient is obtained by arithmetic averaging of the gradients of two neighboring cell plus a solution jump term that enhances the stability of the scheme and does not change the consistency of the discretization [37, 61].

$$\vec{F}_v = \frac{\vec{F}\left(U^L\right) + \vec{F}\left(U^R\right)}{2} + \frac{\alpha}{\overrightarrow{r_{ij}} \cdot \hat{n}} \left(U^L - U^R\right) \hat{n} \tag{1.15}$$

where $\vec{F}\left(U^L\right)$ and $\vec{F}\left(U^R\right)$ are the viscous flux vectors computed at each side and the second term on right hand side is the jump term in which $\overrightarrow{r_{ij}}$ is the vector from the reference point of cell i to its immediate neighbor, cell j and α is a constant.

1.2.4 Solution Method

In this thesis, we look for steady-state solutions for which integration in time of the spatially discretized equation, Eq. 1.5, can be implemented explicitly or implicitly. In explicit time integration, the space discretization is performed at the previous time level using the known flow quantities found at the previous time iteration.

$$\frac{\overline{U}_i^{n+1} - \overline{U}_i^n}{\Delta t} = -R\left(\overline{U}_i^n\right) \tag{1.16}$$

where $\overline{U}_i^n$ is the solution average vector at the current time level and $\overline{U}_i^{n+1}$ is the solution at the next time level. In implicit time integration, both the space and the time discretizations are performed at the current time level where the flow quantities are needed as unknowns.

$$\frac{\overline{U}_i^{n+1} - \overline{U}_i^n}{\Delta t} = -R\left(\overline{U}_i^{n+1}\right) \tag{1.17}$$

Implicit methods do not suffer from the stability restrictions of explicit methods and large time steps can be taken. As a result, faster convergence to the steady-state solution is possible when implicit time-stepping techniques are used. To solve Eq. 1.17 numerically, both sides of the equation should be evaluated at time level $n+1$. This is implemented by backward time differencing of the left hand side and residual linearization in time for the right hand side about the state $\overline{U}_i^n$:

$$\begin{aligned}
\frac{\overline{U}_i^{n+1} - \overline{U}_i^n}{\Delta t} &= -R\left(\overline{U}_i^{n+1}\right) \\
&= -\left[R\left(\overline{U}_i^n\right) + \frac{\partial R}{\partial \overline{U}}\left(\overline{U}_i^{n+1} - \overline{U}_i^n\right) + O\left(\overline{U}_i^{n+1} - \overline{U}_i^n\right)^2\right] \\
\Rightarrow \left(\frac{I}{\Delta t} + \frac{\partial R}{\partial \overline{U}}\right)\delta\overline{U}_i &= -R\left(\overline{U}_i^n\right)
\end{aligned} \tag{1.18}$$

where $\delta\overline{U}_i = \overline{U}_i^{n+1} - \overline{U}_i^n$ and $\frac{\partial R}{\partial \overline{U}}$ is the global Jacobian matrix resulting from flux integral or residual linearization. Eq. 1.18 is a large linear system of equations which needs to be solved at each time step to obtain an update for the vector of unknowns.

The global Jacobian matrix can be represented explicitly as:

$$\frac{\partial R}{\partial \overline{U}} = \frac{\partial FluxInt}{\partial Flux}\frac{\partial Flux}{\partial RecSol}\frac{\partial RecSol}{\partial RecCoef}\frac{\partial RecCoef}{\partial \overline{U}} \tag{1.19}$$

The last term $\frac{\partial RecCoef}{\partial \overline{U}}$ is found by using the pseudoinverse of the reconstruction matrix, $\frac{\partial RecSol}{\partial RecCoef}$ is a geometric term dependent on the location of the Gauss points, $\frac{\partial Flux}{\partial RecSol}$ is based on the flux function and $\frac{\partial FluxInt}{\partial Flux}$ is computed using the appropriate Gauss integration weight and the length of edges [52].

The computational cost, both in terms of memory usage and time, of flux computation, flux integration, Jacobian matrix calculation and other associated numerical calculations per cell increase when higher-order methods are implemented; however, as a result of using a coarser mesh, the overall computational cost to obtain a determined accuracy is less if higher-order methods are used [53, 56, 58, 64].

1.3 Truncation and Discretization Error

There are two error terms describing the accuracy of the scheme: discretization error and truncation error[1]. Discretization error is defined as the difference between the discrete solution obtained by the CFD simulation of the governing equation and the exact solution of the problem. The initial source of these numerical errors is the truncation error, the amount by which the discrete solution fails to satisfy the PDE locally. Assume an exact solution $\tilde{U}$ to a linear differential linear equation $L\left(\tilde{U}\right) = 0$ and its discrete analog $L_h\left(\tilde{U}_h\right) = 0$. We can project the continuous solution onto a mesh with characteristic size h to get $I^h\tilde{U}$ which does not satisfy the discrete equation $L_h\left(I^h\tilde{U}\right) \neq 0$:

$$L_h\left(I^h\tilde{U}\right) = L_h\left(\tilde{U}_h\right) + \frac{\partial L_h}{\partial U_h}\left(I^h\tilde{U} - \tilde{U}_h\right) + O\left(I^h\tilde{U} - \tilde{U}_h\right)^2 \tag{1.20}$$

where the left-hand side is the truncation error, τ, the first term on the right-hand side is zero; $\frac{\partial L_h}{\partial U_h}$ is the flux Jacobian described above and $I^h\tilde{U} - \tilde{U}_h$ is the discretization error, ϵ. This is called the error transport equation which relates the truncation and discretization error. Error transport equation has been used for error estimation [80].

As an example, consider the finite volume discretization of Laplace equation on the one-dimensional grid shown on Figure 1.6:

$$\frac{d^2T}{dx^2} = 0. \tag{1.21}$$

This finite volume discretization of this equation is considered. The averages over

[1]We are not considering the roundoff error and iterative error.

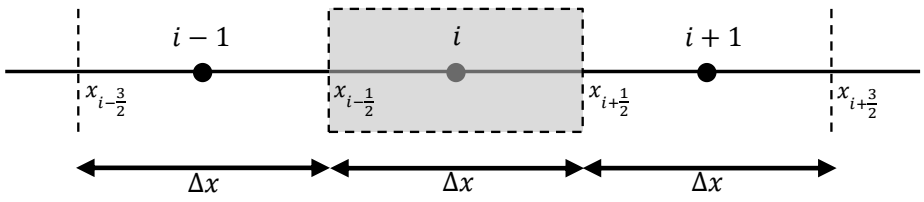


Figure 1.6: Discretization stencil for Laplace equation

control volume i is taken:

$$\overline{\frac{d^2T}{dx^2}} = \frac{1}{\triangle x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \left(\frac{d^2T}{dx^2}\right) dx = \frac{\left.\frac{dT}{dx}\right|_{i+\frac{1}{2}} - \left.\frac{dT}{dx}\right|_{i-\frac{1}{2}}}{\triangle x} \approx \frac{\overline{\left.\frac{dT}{dx}\right|_{i+1}} - \overline{\left.\frac{dT}{dx}\right|_{i-1}}}{2\triangle x} \tag{1.22}$$

Using the Taylor series expansion, the control volume averages are obtained in terms of derivatives at point i:

$$\begin{aligned}
\overline{\left.\frac{dT}{dx}\right|_{i+1}} &= \frac{1}{\Delta x}\int_{x_{i+\frac{1}{2}}}^{x_{i+\frac{3}{2}}} \\
&\quad \left(\left.\frac{dT}{dx}\right|_i + \left.\frac{d^2T}{dx^2}\right|_i (x-x_i) + \frac{1}{2}\left.\frac{d^3T}{dx^3}\right|_i (x-x_i)^2 + \frac{1}{6}\left.\frac{d^4T}{dx^4}\right|_i (x-x_i)^3 \ldots\right) dx \\
&= \left.\frac{dT}{dx}\right|_i + \left.\frac{d^2T}{dx^2}\right|_i (\Delta x) + \frac{13}{24}\left.\frac{d^3T}{dx^3}\right|_i (\Delta x)^2 + \frac{5}{24}\left.\frac{d^4T}{dx^4}\right|_i (\Delta x)^3 + \mathcal{O}(\Delta x)^4
\end{aligned}$$

$$\begin{aligned}
\overline{\left.\frac{dT}{dx}\right|_{i-1}} &= \frac{1}{\Delta x}\int_{x_{i-\frac{3}{2}}}^{x_{i-\frac{1}{2}}} \\
&\quad \left(\left.\frac{dT}{dx}\right|_i + \left.\frac{d^2T}{dx^2}\right|_i (x-x_i) + \frac{1}{2}\left.\frac{d^3T}{dx^3}\right|_i (x-x_i)^2 + \frac{1}{6}\left.\frac{d^4T}{dx^4}\right|_i (x-x_i)^3 \ldots\right) dx \\
&= \left.\frac{dT}{dx}\right|_i - \left.\frac{d^2T}{dx^2}\right|_i (\Delta x) + \frac{13}{24}\left.\frac{d^3T}{dx^3}\right|_i (\Delta x)^2 - \frac{5}{24}\left.\frac{d^4T}{dx^4}\right|_i (\Delta x)^3 + \mathcal{O}(\Delta x)^4
\end{aligned}$$

Substituting these control volume averages into Eq. 1.22 gives the finite volume ap-

proximation of the second derivative as:

$$\overline{\frac{d^2T}{dx^2}} = \left.\frac{d^2T}{dx^2}\right|_i + \frac{5}{24}\left.\frac{d^4T}{dx^4}\right|_i (\Delta x)^2 + \mathcal{O}(\Delta x)^4 \tag{1.23}$$

in which $\frac{5\Delta x^2}{24}\left.\frac{\partial^4 T}{\partial x^4}\right|_i$ is the leading order term in the truncation error.

Historically, truncation error analysis for structured mesh schemes is a routine application of Taylor series analysis, while analysis of unstructured mesh schemes has lagged behind. Recent work on analysis of unstructured mesh schemes includes verification of unstructured mesh solvers [91], accurately predicting flow properties on unstructured meshes [14], numerical studies of error for mixed element finite-volume schemes [17], accuracy of discretization schemes on irregular grids [18], error comparisons for cell-centered or vertex-centered discretizations [20], and Taylor-based accuracy assessment of cell-centered discretizations [36].

Local analysis of truncation error is needed as some local mesh configurations may result in asymptotically larger errors than others. Mesh features, including cell size, anisotropy, shape and connectivity, can have an adverse interaction with discretization schemes that affects the solution accuracy. Error quantification hinges on the ability to accurately estimate and efficiently exploit the local truncation error.

Ultimately, we need information about configurations in a mesh producing large error locally, which implies a need for local analysis of truncation error for arbitrary cells. Most truncation error estimation schemes produce estimates of the overall truncation error on a cell-by-cell basis. Ollivier-Gooch and Van-Altena [66] performed Taylor series truncation error analysis for the Laplacian on regular triangular meshes, and this was extended to general stencils by Jalali and Ollivier-Gooch [36]. The output of this analysis is a set of Taylor series coefficients for the truncation error for each control volume which can be combined with local solution derivatives to compute the truncation error.

Several researchers, including Diskin and Thomas [17, 19], and Jalali and Ollivier-Gooch [36], have demonstrated that the truncation error for unstructured finite volume schemes is asymptotically larger than the discretization error which in turn is typically of the same order as the solution approximation error in the scheme. This behavior is in contrast with the structured mesh case for which the truncation error has the same asymptotic order of accuracy as the discretization error.

Another feature of the truncation error for unstructured mesh schemes is its noisy

appearance, caused by the discontinuous jump of the coefficients of the terms in the Taylor series expansion of the error from one control volume to another. This is in contrast with structured mesh schemes where the truncation error is smooth. These two features of the truncation error for unstructured mesh finite volume schemes, non-smoothness and large magnitude, are related to each other by the eigensystem of the discrete problem as shown by Ollivier-Gooch and Roy [65]. Sharbatdar and Ollivier-Gooch [84] have shown by eigenanalysis of the truncation error that the rough modes dominate the unstructured mesh truncation error. Details of eigenanalysis of the truncation and discretization error will be described in Chapter 2.

1.4 Truncation Error Estimation Methods

Consider a steady partial differential equation with exact solution of $\tilde{U}$:

$$L\left(\tilde{U}\right) = 0 \tag{1.24}$$

where L is a partial differential operator. Let L_h be a discrete approximation to L on a mesh with characteristic size h and $\tilde{U}_h$ denote the exact solution to the discrete problem. The discrete form of Eq. 1.24 is written as:

$$L_{h,p}\left(\tilde{U}_{h,p}\right) = 0 \tag{1.25}$$

where the original discrete problem is solved to order p. The exact solution, $\tilde{U}$, when projected onto the mesh, does not satisfy the discrete equation:

$$L_{h,p}\left(I^h\tilde{U}_p\right) \neq 0 \tag{1.26}$$

where I^h denotes a restriction operator that projects a continuous function onto the discrete space. Roy [80] showed that the truncation error, $\tau_{h,p}(U)$, can be defined as:

$$L_{h,p}\left(I^hU\right) = I^hL(U) + \tau_{h,p}(U) \tag{1.27}$$

in which U is any continuous function.

Quantifying the truncation error in the sense of finding corresponding coefficients to each spatial derivative is well understood for one-dimensional discretization using either finite difference, finite element or finite volume methods. Extending the analysis to the

multi-dimensional Cartesian discretization is straightforward as well. However, the truncation error analysis of discretization schemes being used for unstructured meshes is more difficult because the local shape and connectivity of the mesh is more varied. Besides, for non-linear equations, finding the coefficients in the Taylor series expansion is very complicated. As a result, other methods are required for obtaining an estimation of the truncation error. In this section, a number of common methods for estimating the truncation error are considered.

Truncation error estimation in the finite element context has been throughly studied; see [3] for the seminal discussion. These methods have been extended to the Discontinuous Galerkin framework [2, 41]. Most of these methods work better for elliptic problems than for hyperbolic problems. Moreover, discontinuities, singularities, and geometrical complexities can significantly reduce the reliability of these error estimation methods. In finite elements, using the same mesh with different orders of accuracy is an alternative for estimating the truncation error. Mavriplis [50] estimated the local error by measuring the energy norm associated with the different method. Finite element residual methods make use of the continuous representation provided by the approximate finite element solution to evaluate a continuous representation of the residual [3].

In the finite volume context, there are several options for estimating the truncation error including using the exact solution, using the solution on a finer mesh, and using a higher order discretization scheme.

1.4.1 Exact Truncation Error

When $\tilde{U}$, the exact solution of the PDE, is available, then the the first term on the right-hand side in Eq. 1.27 is zero and the p-order truncation error is simply:

$$L_{h,p}\left(I^h\tilde{U}\right) = \tau_{h,p}\left(\tilde{U}\right) \tag{1.28}$$

For finite-volume methods, this implies taking control volume averages of the exact solution and computing a flux integral based on that. Even though calculating the truncation error by this approach provides us with the exact truncation error, applying the method to general problems with unknown exact solution is not possible. However, this approach may be utilized as a reference value to compare other estimations with.

1.4.2 Estimating Truncation Error using a Finer Mesh

Instead of the exact solution of the PDE, the discrete solution on a finer mesh with characteristic size $\frac{h}{2}$ may approximate the exact solution. When the solution is in the asymptotic range and the scheme is $p-$order accurate, the residual on a finer mesh $L_{\frac{h}{2}}\left(I_h^{h/2}U_h\right)$ will be a factor of 2^p smaller than the truncation error. This method, called the h-truncation error estimate, or h-TE, is used locally in finite element adjoint-based adaptation schemes by Darmofal *et al.* [94, 95] for both viscous and inviscid flows, Nemec and Aftosmis [59, 60] and Hartmann [48]. However, for a global truncation error estimate, the requirement that the solution be in the asymptotic range for truncation error to be correct is undesirable. Furthermore, the requirement of the residual computation on the fine mesh implies more computational cost in terms of time and memory compared to a single mesh method.

1.4.3 Estimating Truncation Error using a Higher-Order Scheme

Another approach for estimating the truncation error which is not limited to problems with exact solution uses a higher order discretization to find the leading order terms in the truncation error estimate, producing the p-truncation error estimate, or p-TE. The residual computed to p-order in Eq. 1.25 is necessarily zero. If we used the p-order solution in a $p+1$ order discretization, we will get the leading order terms in the truncation error of the p-order scheme:

$$L_{h,p+1}(U_{h,p}) = \tau_{h;p\to p+1}\left(U_{h,p}\right) \tag{1.29}$$

The accuracy of this estimate could be improved by using even higher order discretizations, $p+2$ instead of $p+1$, and adding the next higher order terms to the truncation error estimates. This scheme was used for optimizing an unstructured mesh to reduce the truncation error [65].

1.5 Application of Truncation Error Estimation

Error quantification hinges on the ability to accurately estimate and efficiently exploit the local truncation error. The truncation error estimate can be used directly to improve the solution through defect correction. Furthermore, it can be combined with the

solution to an auxiliary partial differential equation, the adjoint problem, to estimate the error in an output quantity of engineering interest [26]. Another important application of the estimation of the truncation error is to drive mesh adaptation to improve the efficiency of the simulation.

1.5.1 Defect Correction

Over the years, there have been numerous attempts to estimate the error in discrete solutions or, equivalently, to improve the accuracy of the solution. Two widely applicable approaches are Richardson extrapolation and defect correction. Although both approaches are used in an iterative fashion, defect correction proceeds on the original grid of the discretization while Richardson extrapolation needs repeated grid refinement and yet produces answers on the original grid only.

Defect correction methods are based on a particular way to estimate local or global errors. The use of simple integration schemes in combination with defect (residual) evaluation leads to computable error estimates and, in an iterative fashion, yields improved numerical solutions. Approximating the residual by discretization is the standard approach for more traditional uses of defect correction [73, 87] and requires the use of higher-order discretization than used in the primal equation.

Consider the a steady partial differential equation with source term f as:

$$L\left(U\right) = f \tag{1.30}$$

The discrete form can be written as:

$$L_h\left(U_h\right) = 0$$

and the amount by which the discrete solution fails to satisfy the PDE is the exact truncation error as

$$I^h f - L_h\left(I^h U\right) = \tau$$

So the original discrete equation is

$$L_h\left(I^h U\right) = I^h f - \tau \tag{1.31}$$

which implies that if the truncation error is added to the source term of the problem,

Eq. 1.30,

$$L(U) = f + \tau \tag{1.32}$$

the exact solution is obtained, referring back to Eq. 1.31.

$$L_h(U_h) = I^h f + \tau - \tau = I^h f.$$

$$L_h(U_h) = I^h f \tag{1.33}$$

In the defect correction procedure, the higher order discrete operator is applied to the lower order solution [12, 24, 29, 49, 69, 70, 86, 87] instead of the exact truncation error. This defect correction method provides local discretization error estimates driven by the p-TE estimate based on higher order discretization and treats the truncation error estimate as a source term to drive the original problem toward a higher-order solution. Theoretically, adding the exact truncation error to the source term of the problem gives us the exact solution, and adding the p-TE estimate to the source term gives us the solution which converges to the exact solution at p-order.

1.5.2 Output Error Estimation

In the field of computational aerodynamics and fluid dynamics, lift and drag are output functionals of interest and the desire for efficient computational methods producing reliable and accurate lift and drag values drives algorithm research in the field. The adjoint method has played an important role in this context because of the great flexibility it offers with regard to the physics model and to the definition of output functionals. The history of the use of adjoint equations in fluid dynamics design goes back to the work by Pironneau [74] and particularly in the field of computational aerodynamics design to the work by Jameson [39]. Since then, adjoint methods have been used for design applications for both internal and external flows [38, 39, 43, 44, 54, 75]. The adjoint theory was first presented in the context of linear algebra by using the algebraic equations obtained from the discretization of the original problem. This is the basis for the discrete adjoint approach. The continuous adjoint approach, on the other hand, is formulated based on the adjoint PDE which is discretized and solved independently [26].

Consider a partial differential equation arising from a finite volume discretization

of the original fluid dynamic equations which is discretized and written as an algebraic equation:

$$R_h\left(\overline{U}_h\right) = 0. \tag{1.34}$$

For a scalar output, $J_h\left(\overline{U}_h\right)$ such as lift or drag, the associated discrete adjoint vector, Z_h, must satisfy the discrete adjoint equation:

$$\left(\frac{\partial R_h}{\partial \overline{U}_h}\right)^T Z_h + \left(\frac{\partial J_h}{\partial \overline{U}_h}\right)^T = 0. \tag{1.35}$$

As discussed in Section 1.2, the implicit Jacobian matrix can be computed in our solver and hence the transpose of the Jacobian matrix, $\left(\frac{\partial R_h}{\partial \overline{U}_h}\right)^T$, can be calculated easily.

The continuous primal problem can be posed as

$$\begin{array}{lrcll} \textit{Determine} & J & = & (U,g)_D + (CU,h)_{\partial D} & \\ \textit{given that} & LU & = & f & \textit{in } D \\ & BU & = & e & \textit{in } \partial D \end{array} \tag{1.36}$$

where $(U,V)_D = \int_D U^T V dv$ is an integral over the domain and $(U,V)_{\partial D} = \int_{\partial D} U^T V dA$ is an integral over the boundary of the domain. The objective in the analytic approach is to convert the primal problem using integration by parts into an equivalent adjoint problem

$$\begin{array}{lrcll} \textit{Determine} & J & = & (Z,f)_D + (C^*Z,e)_{\partial D} & \\ \textit{given that} & L^*Z & = & g & \textit{in } D \\ & B^*Z & = & h & \textit{on } \partial D \end{array} \tag{1.37}$$

where L^* is the linear PDE which is adjoint to L. B and B^* are boundary condition operators for the primal and adjoint problems, respectively, and C and C^* are possible differential weight boundary operators used in the functionals; these four operators may have different dimensions on different parts of the boundary. The two forms of the problem are equivalent provided that

$$J = (Z,LU)_D + (C^*Z,BU)_{\partial D} = (L^*Z,U)_D + (B^*Z,CU)_{\partial D} \tag{1.38}$$

which is the adjoint consistency condition. The left hand side is the output functional based on the adjoint solution and the right hand side is the output functional based on the primal problem.

The adjoint problem plays a key role in estimating and reducing the error in output

functional. Within the context of finite element methods, output functional correction has been outlined by Becker and Rannacher [10] and Larson and Barth [47] based on structural finite element methods [3]. The adjoint-based error correction technique developed by Pierce and Giles [72] extends the inherent super-convergence properties of finite element methods to cover numerical results obtained from finite difference and finite volume without natural super-convergence properties. Moreover, the technique can be used to improve the accuracy of super-convergent functionals obtained from finite element methods by constructing smoother, higher-order interpolants of the primal and adjoint solutions [72].

It is easy to show that the exact error in the output functional can be written as:

$$J = (U, g) = (U_h, g_h) - (Z_h, \tau_h) + H.O.T \tag{1.39}$$

where $U - U_h$ is the discretization error and the higher order terms ($H.O.T$) involves the unknown discretization error in the PDE and the truncation error for the adjoint problem.

For numerical efficiency, computing the functional value to a higher order of accuracy than the primal solution is advantageous. A super-convergent estimate of the functional may be obtained by computing the leading error term in the original functional estimate and using this as a correction. Pierce and Giles [73] showed that the error in the functional value based on the reconstructed primal solution can be expressed as a function of the truncation error, implying that truncation error estimation is required for output error estimation. This error estimation technique has been exploited for large-scale CFD simulations in the discontinuous Galerkin community [23, 31, 32, 33, 59, 68, 94, 95], where higher asymptotic convergence rates are routinely obtained. Venditti and Darmofal presented an error estimation strategy based on adjoint formulation for estimating errors in functional outputs for one and two-dimensional problems and their error estimation procedure was applied to a standard, second-order finite volume discretization [93]. The h-TE method is used for estimating the truncation error, the solution on a finer mesh, and the discrete adjoint solution.

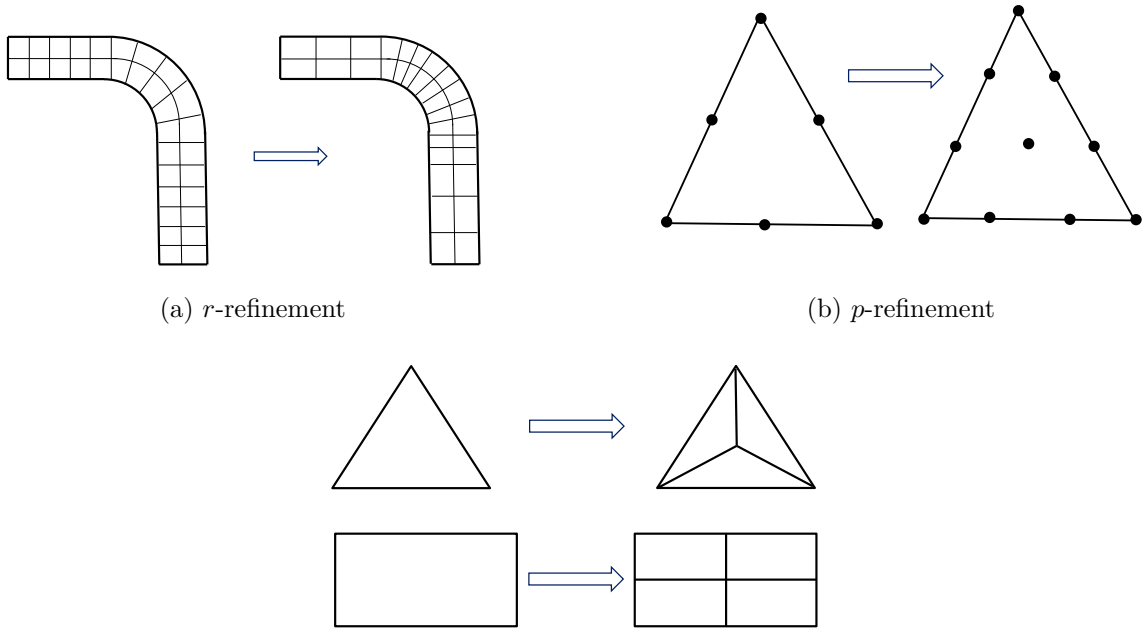
More details of obtaining the adjoint solution and correcting the output functional will be described in Chapter 5.

1.5.3 Mesh Adaptation

The accuracy of the final solution is highly dependent on the mesh spacing. Accordingly, one approach for achieving a more accurate solution is to repeat the problem with increasingly finer meshes until an acceptable variation, i.e. within some tolerance from one solution to another is obtained; however, this approach is inefficient and time consuming. Assessing and minimizing discretization error by this method requires time consuming mesh dependence analysis or computation of solutions on meshes that are finer than the required solution accuracy dictates. Furthermore, a local difference in element shape, orientation and size can influence the solution. Hence, mesh refinement, especially for unstructured meshes, is only asymptotically guaranteed to produce a more accurate solution.

One method employed to overcome this problem is mesh adaptation, in which the mesh is locally fit to the particular features of the flow. Mesh adaptation strategies can usually be classified as one of the three general types: $r-$refinement, $p-$refinement, and $h-$refinement. $r-$refinement is modification of the mesh resolution, without changing mesh connectivity or the number of vertices in the mesh. The mesh vertices in $r-$refinement are moved such that the density of points in particular regions of the mesh with more interesting flow features such as boundary layers increases while the number of mesh points in other regions decreases, Figure 1.7a. Increased resolution is achieved by increasing the order of accuracy of the polynomials in each element if $p-$refinement is implemented. This is depicted in Figure 1.7b for a nodal finite element scheme, increasing order of accuracy from second to third for a triangular element. $h-$adaptation decreases the mesh spacing directly by splitting the cell into cells with smaller size, Figure 1.7c. I will use $h-$refinement in this thesis, so adaptation implies $h-$refinement throughout this thesis.

Mesh adaptation seeks to automate the process of minimizing discretization error by first computing the solution on a coarse mesh and then successively refining the mesh so that the optimal mesh for the solution being computed is produced [81, 83]. A mesh generator should aim at achieving the best possible mesh using as few mesh points as possible since the computational costs (both in terms of memory usage and time) increases with the number of elements in the mesh. It is useful to make the mesh finer in certain regions where the solution details must be captured, near boundaries for instance, while keeping it coarse everywhere else (such as far fields). One of the most important properties of a proper mesh for numerical simulations is that the density



(a) r-refinement

(b) p-refinement

(c) h-refinement

Figure 1.7: Mesh adaptation strategies

of mesh points should be easily controllable and allowed to vary quickly over a short distance [27]. This property, which is easily available in adaptive schemes, saves simulation time since it prevents time from being wasted in regions that would otherwise be over-resolved.

Several different adaptation indicators may be exploited to drive mesh adaptation. Feature-based grid adaptation uses solution gradient, solution curvature, or even identified solution features to drive the adaptation process. This commonly-used approach relies on a priori knowledge of the flow, including whether the solution contains shock waves, boundary layers, etc., as well as a reliable feature-detection system. The main advantage of explicitly detecting features is that more specific refinement is possible. On the other hand, feature-based adaptation may result in over-refined features while some other features are not refined enough [79]. In some cases, feature-based refinement can actually increase the solution error [3, 79]. Dwight [22] has shown the failure of feature-based adaptation for the inviscid transonic flow over an airfoil using an unstructured finite-volume discretization. Adaptation based on solution gradients initially shows a reduction in the discretization error, but then subsequent adaptation steps show an increase in the discretization error.

Using local solution approximation error as an error indicator has had some success. These error estimators identify locations where the solution is not well represented on the mesh. This typically implies estimating the lowest-order solution derivatives that are not directly computed during the solution process and using these to create a metric definition for the solution [4, 13, 21, 28, 67]. The mesh is then adapted to match the metric. This approach has recently been extended for use with high-order methods [67].

Because truncation error, the amount by which the discrete solution fails to satisfy the PDE locally, is the source term in the error transport equation for discretization error, using truncation error to drive adaptations adds degrees of freedom to a simulation precisely where additional accuracy will reduce discretization error directly [79]. Small errors from one location of the mesh can be convected to regions which are relatively far away since the error in values used in the Taylor approximation are passed on to the interpolated value which will affect other Taylor expansion terms and so on [67]. Those PDEs containing strong convective terms, such as the Euler equations, suffer from this drawback of the residual-based error estimation. Accordingly, the disadvantage of using this method is that features may be detected in the wrong location as a consequence of convected errors.

Adjoint-based mesh adaptation is another common adaptation indicator [8, 31, 73, 93, 95]. It was shown in the previous section that the error in an output functional can be expressed as an inner product of the local residual errors and the adjoint solution. By estimating the local contribution of each cell to the error in any solution functional of interest, this approach localizes mesh refinement to those parts of the computational domain that most influence the accuracy of the selected outputs [60] and so optimally reduces error in an output functional. Becker and Rannacher [11] have developed this output-based adaptive procedure by exploiting finite element orthogonality properties. This approach has been extended to the finite volume method; for example, Venditti and Darmofal [94] successfully applied this approach for inviscid and viscous flow over airfoils solved to second-order accurate. For situations where the only desired output is a function of the solution, adjoint-based methods are ideal because they do not refine areas of the mesh where the solution has no effect on that function. Analyzing the drag force on an airfoil is an example; the mesh created by adjoint-based refinement provides more accurate drag prediction than other meshes of similar size as will be shown in Chapter 6. Adjoint-based mesh adaptation is widely used for all finite difference [45], finite element [23, 30, 85, 97, 98] and finite volume [16, 25] methods.

1.6 Objectives

The goal of this thesis is to develop and benchmark a reliable truncation error estimate for finite-volume schemes on unstructured meshes and to use this truncation error estimate to improve the solution through defect correction, to estimate solution functional error, and to drive mesh adaptation.

The commonly-used method for estimating the truncation error on structured meshes is using a higher-order discretization to find the leading order term in the truncation error, p-TE method. However, there is no guarantee that the performance of the p-TE method is the same for unstructured meshes. Eigenanalysis is used as a tool to study the behavior of the truncation error on unstructured meshes. If using the original p-TE based on the discrete solution is not sufficient for our purposes when unstructured mesh is used, we need to develop a modified estimation of the truncation error that can be computed using quantities available in a typical second-order finite volume solver.

We will investigate the effectiveness of the developed estimation of the truncation error for application to defect correction, output functional error estimation, and mesh adaptation. In all cases, the success of our truncation error estimate will be its effectiveness in reducing error is the solution and output functional.

As test cases, we will focus initially on two-dimensional problems with known, manufactured solutions. In this context, we use the exact solution for computing discretization error and functional output error and to produce a reference value of truncation error by computing the flux integral from control volume averages of the exact solution. We will be applying this technique to two-dimensional compressible flow problems. Because these techniques are mathematical in origin rather than physical, subsequently applying them to other physical problems will not be difficult.

1.7 Outline

An overview of the eigenanalysis of truncation error is described in Chapter 2 and the smoothness of the truncation error on unstructured meshes is compared to the structured meshes. As rough modes of the truncation error are dominant for unstructured meshes, a smooth estimation of truncation error is required.

Chapter 3 is devoted to the solution interpolation techniques. We originally hope to get a smooth estimation of truncation error. For this purpose, we developed an estimation of the truncation error based on the interpolation of the solution. Two

different interpolation schemes are described in this chapter, CGM and C^1 interpolation. CGM is a code library providing geometry functionality for mesh generation, and C^1 interpolation approximates the reconstructed finite volume solution with continuity along the edges between elements of a triangulation. Although the estimate of the truncation error based on either one does not have a smooth distribution, it will be shown that the truncation error estimate based on C^1 interpolated solution leads to similar behavior obtained by smooth truncation error on structured meshes.

The general algorithm of defect correction is described in Chapter 4 and the performance of defect correction with discrete p-TE and continuous p-TE is compared for scalar problems of Poisson and advection with manufactured solution.

Chapter 5 is devoted to output error estimation and correction and the different functionals are corrected based on the continuous p-truncation error multiplied by both continuous and discrete adjoint solutions. Scalar model problems and inviscid and viscous compressible flow problems are considered.

In Chapter 6, the performance of the continuous p-truncation error as adaptation indicator is compared to the discrete p-TE estimate. As described, using truncation error to drive adaptation adds degrees of freedom to a simulation precisely where additional accuracy will reduce discretization error directly and a variation on this approach is to weight the truncation error with the adjoint solution, either continuous or discrete adjoint solution. Adaptation based on the weighted truncation error reduces error in an output functional by refining the mesh in places where the PDE is poorly solved and that error has a large impact on the output of interest. The performance of these weighted truncation error indicators are also compared to the truncation error itself as an error indicator.

The thesis concludes in Chapter 7 with a summary of the research, contributions, conclusions based on the results and recommendations for future work.

Chapter 2

Eigenanalysis of the Truncation Error

The behavior of truncation error for unstructured mesh solvers is qualitatively different from structured meshes. Roe studied the behavior of the local truncation error for second-order accurate scheme on a general unstructured meshes and he showed that the local truncation error is not necessarily second-order accurate [78]. Consider an unstructured discretization of the Laplacian, for instance. A typical second-order accurate scheme will compute a first order accurate gradient for which the flux integration results in a zero-order residual. Second-order accurate solutions are obtained as a result of cancellation of the contribution of these large local errors to the overall solution. On a structured mesh, on the other hand, as a consequence of smoothness and symmetry in the mesh, cancellation of errors leads to second order flux integrals. Hence, the behavior of truncation error on structured and unstructured mesh are completely different.

In this chapter, we will use eigenanalysis of the Jacobian matrix as a tool to investigate the truncation error distribution on unstructured meshes. As truncation error is the source of discretization error, the discretization error distribution is also different from structured meshes. Eigenanalysis of truncation error is done for two test cases: one model problem, the Poisson equation with a manufactured solution, and an inviscid compressible flow problem.

2.1 Truncation and Discretization Error on Structured and Unstructured Meshes

The solution and both the discretization and truncation errors have smooth distributions when computed using structured meshes because symmetry and smoothness in the mesh leads to cancellation in the error. The truncation error can easily be derived using Taylor series analysis [6, 99]. On the other hand, the truncation error, and conse-

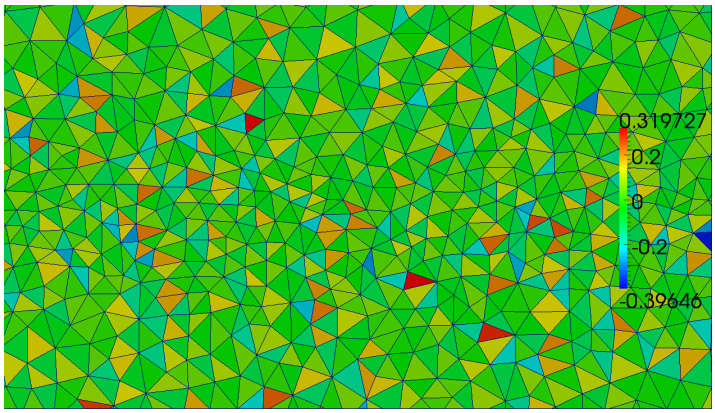


Figure 2.1: Coefficient of error term in the leas-squares gradient on a general unstructured mesh for Poisson, λ_{xx} coefficients

quently the discretization error, for unstructured mesh schemes has a noisy appearance, caused by the discontinuous jump of the coefficients of the terms in the Taylor series expansion of the error from one control volume to another. Taylor series truncation error analysis can be extended to any arbitrary stencil in an unstructured mesh [35]. The truncation error for the Poisson equation can be written as

$$\tau = h^0 \left(\lambda_{xx} \frac{\partial^2 T}{\partial x^2} + \lambda_{xy} \frac{\partial^2 T}{\partial x \partial y} + \lambda_{yy} \frac{\partial^2 T}{\partial y^2} \right) + O\left(h\right) \tag{2.1}$$

as described in Jalali's thesis [35]. Figure 2.1 depicts one of the coefficients, λ_{xx}, in the Taylor series expansion for each cell. As shown in this figure, the coefficients are different for neighboring cells. In this section, the different truncation error distributions are illustrated for structured and unstructured meshes.

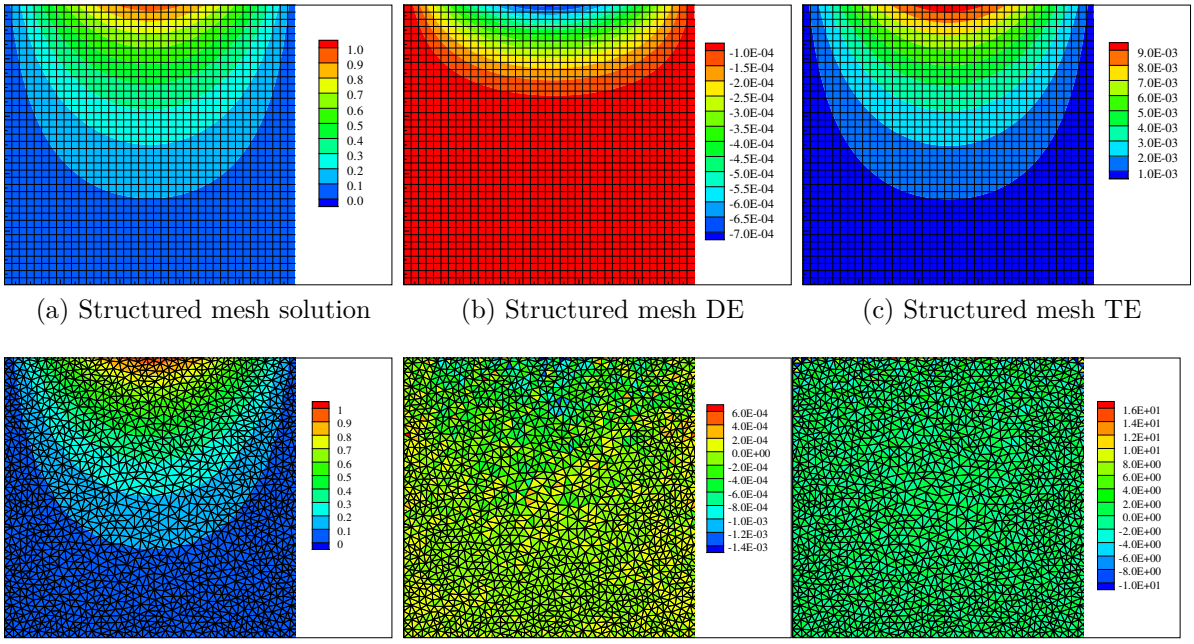
2.1.1 Error Distribution on Meshes

Consider the Poisson problem in a (1×1) square domain with Dirichlet boundary conditions. The manufactured solution [62] is set as

$$U = \frac{\pi}{8} \sin\left(\pi x\right) \sin\left(\pi y\right) + \frac{1}{\sinh\left(\pi\right)} \sin\left(\pi x\right) \sinh\left(\pi y\right)$$

for which the source term is obtained:

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = -\frac{\pi^3}{4} \sin\left(\pi x\right) \sin\left(\pi y\right) \tag{2.2}$$



(a) Structured mesh solution (b) Structured mesh DE (c) Structured mesh TE

(d) Unstructured mesh solution (e) Unstructured mesh DE (f) Unstructured mesh TE

Figure 2.2: Comparison of truncation and discretization error for structured and unstructured meshes for Poisson in 2D

The problem is solved to second order accuracy for both structured and unstructured mesh with about the same number of degrees of freedom, for which the solutions are shown in Figure 2.2a and 2.2d, respectively. As a manufactured solution is used, the exact solution is known and the discretization error, the difference between the discrete solution and the exact solution, can be calculated easily. Similarly, the second-order control volume averages may be computed based on the exact solution and used to compute the exact second-order truncation error.

The discretization error is rough for the unstructured mesh, Figure 2.2e, but the maximum error magnitude is reasonable in comparison with the discretization error on structured mesh as depicted in Figure 2.2b.

The truncation error on the structured mesh, Figure 2.2c, is smooth and although it is larger than the discretization error, is still small, whereas on the unstructured mesh, Figure 2.2f, the truncation error is quite noisy, with a peak value three orders of magnitude larger than on the structured mesh. This is the expected behavior for the second-order finite-volume discretization of the Laplacian on unstructured meshes as shown by Jalali and Ollivier-Gooch [36]. To illustrate this difference, consider the flux

integral for the Laplacian:

$$FI = \frac{1}{A} \oint \nabla U ds. \tag{2.3}$$

where the order of ds is the mesh size, $ds \sim h$, and $\frac{1}{A} \sim \frac{1}{h^2}$. For a structured mesh, the gradient can be evaluated using central differences, yielding a second order accurate gradient for which the error varies smoothly. Therefore, in finding the second order accurate flux integral, additional cancellation is obtained. For an unstructured mesh, however, the gradient is evaluated by least-square reconstruction, which yields a first-order gradient with non-smooth error. So, with no cancellation, a zero order flux integral is obtained.

2.1.2 p-TE Method on Unstructured Meshes

Among the different methods for estimating the truncation error, the p-TE method which uses higher order flux integral is one of easiest to apply. We can solve the original discrete problem to order p, where the residual computed to this accuracy is necessarily zero:

$$L_{h,p}\left(U_{h,p}\right) = 0. \tag{2.4}$$

$L_{h,p}$ is a discrete approximation to a linear partial differential operator L with order of accuracy p on a mesh with characteristic size of h. For estimating the truncation error, a higher order discretization is used to find the leading order terms in the truncation error estimate:

$$L_{h,p+1}(U_{h,p}) = \tau_{h;p\to p+1}\left(U_{h,p}\right). \tag{2.5}$$

If accuracy order is very large, $p \to \infty$, the continuous estimation of the truncation error is obtained. Conversely, consider that the p-order operator is applied to the higher-order solution which is a proxy for the continuous solution.

$$L_{h;p}\left(U_{h;p+1}\right) = L_{h;p}\left(U_{h;p} + \delta U\right) \tag{2.6}$$

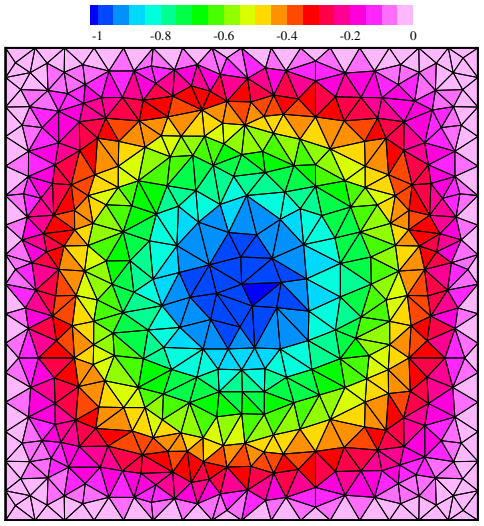


Figure 2.3: The Poisson solution

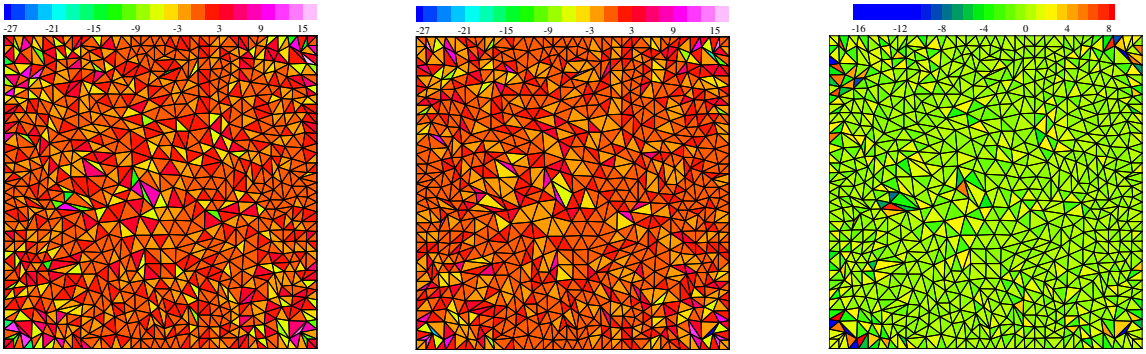
where δU is the difference between the two solutions, $\delta U = U_{h,p+1} - U_{h,p}$. Eq. 2.6 can be expanded by using the linearity of the operator as:

$$\begin{aligned} L_{h;p}\left(U_{h;p+1}\right) &= \left(L_{h;p+1} - \left(L_{h;p+1} - L_{h;p}\right)\right)\left(U_{h;p} + \delta U\right) \\ &= L_{h;p+1}\left(U_{h;p+1}\right) - \left(L_{h;p+1} - L_{h;p}\right)\left(U_{h;p} + \delta U\right) \\ &= 0 - \left(L_{h;p+1} - L_{h;p}\right)\left(U_{h;p}\right) - \left(L_{h;p+1} - L_{h;p}\right)\left(\delta U\right) \\ &= -L_{h;p+1}\left(U_{h;p}\right) + L_{h;p}\left(U_{h;p}\right) - L_{h;p+1}\left(\delta U\right) + L_{h;p}\left(\delta U\right) \\ &= -L_{h;p+1}\left(U_{h;p}\right) + 0 - L_{h;p+1}\left(\delta U\right) + L_{h;p}\left(\delta U\right). \end{aligned} \tag{2.7}$$

For a structured mesh, as a result of symmetry and smoothness in the mesh, the problem is well-behaved and the last two terms in the above equation are of order of $p+1$ and p, respectively. Hence, for a structured mesh,

$$\begin{aligned} L_{h;p}\left(U_{h;p+1}\right) &= -L_{h;p+1}\left(U_{h;p}\right) - O\left(h^{p+1}\right) + O\left(h^{p}\right) \Rightarrow \\ L_{h;p}\left(U_{h;p+1}\right) &= -L_{h;p+1}\left(U_{h;p}\right) + H.O.T. \end{aligned} \tag{2.8}$$

where the left hand side is the leading-order term in the exact truncation error in the sense of being the truncation error associated with the leading-order term in the discretization error, and the right-hand side is a higher-order flux integral based on the p^{th}-order solution. However, for the unstructured mesh with no smoothness in the



(a) L_3U_2 (b) L_2U_3 (c) $L_3U_2+L_2U_3$

Figure 2.4: $L_pU_{p+1} + L_{p+1}U_p$ for an unstructured mesh for Poisson problem of Figure 2.3

truncation error, the orders of the last two terms in Eq. 2.7 are not the same as for a structured mesh. Numerical experiments show that by applying the linear operator on δU, we lose orders in computing these terms. For the Laplacian, for instance, two orders are lost and consequently:

$$\begin{aligned} L_{h;p}\left(U_{h;p+1}\right) &= -L_{h;p+1}\left(U_{h;p}\right) + L_{h;p}\left(U_{h;p}\right) - L_{h;p+1}\left(\delta U\right) + L_{h;p}\left(\delta U\right) \\ &= -L_{h;p+1}\left(U_{h;p}\right) + 0 - O\left(h^{p-1}\right) + O\left(h^{p-2}\right) \\ &= -L_{h;p+1}\left(U_{h;p}\right) + O\left(h^{p-2}\right) \end{aligned} \tag{2.9}$$

Although the sum of the last two terms in Eq. 2.8 are close to zero for a structured mesh, $L_pU_{p+1} + L_{p+1}U_p \neq 0$ for unstructured mesh according to Eq. 2.9 as illustrated in Figure 2.4. Consider the Poisson problem in a (1×1) square, with homogeneous Dirichlet boundary condition with the manufactured solution

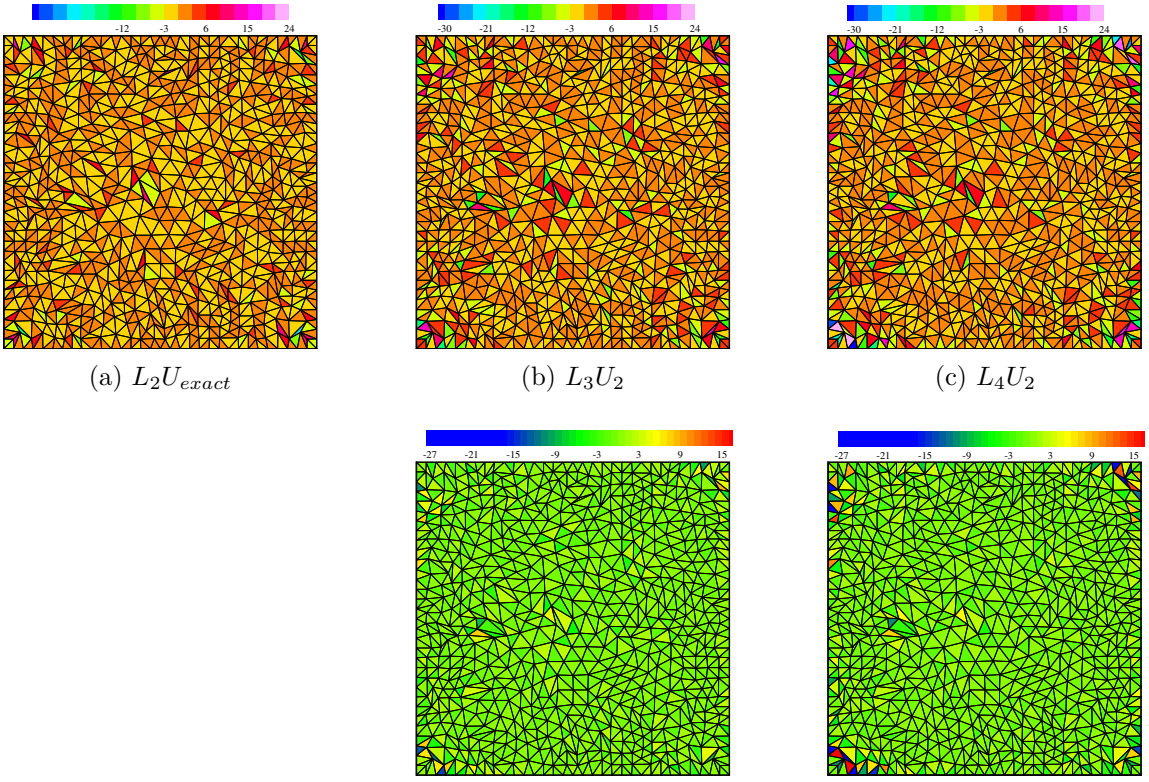
$$U = -\sin\left(\pi x\right)\sin\left(\pi y\right)$$

for which the source term is obtained as:

$$f = 2\pi^2 \sin\left(\pi x\right)\sin\left(\pi y\right)$$

Figure 2.3 shows the solution inside the domain.

Knowing this difference for the unstructured mesh, the accuracy of p-TE method is investigated here. Since the exact solution to this Poisson problem is known, the



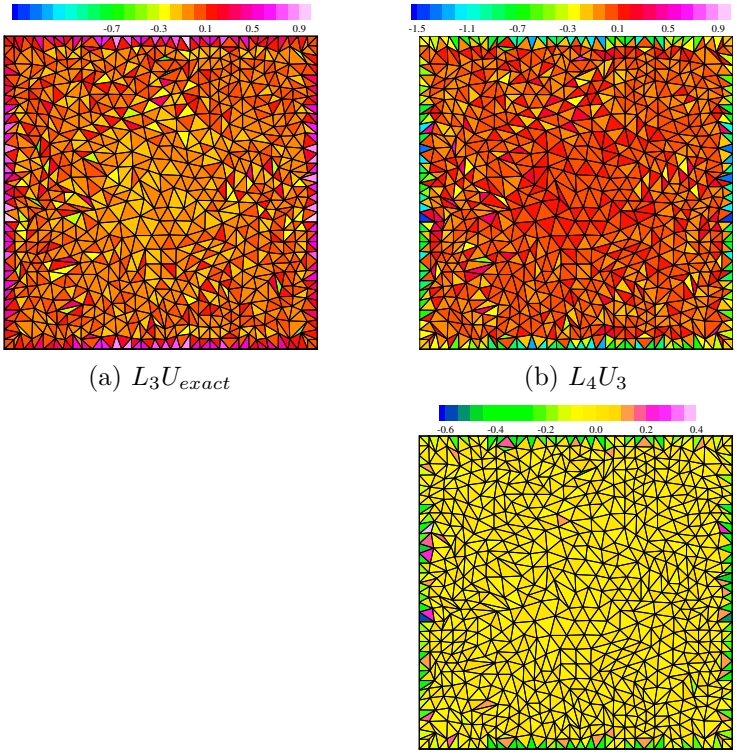
(a) L_2U_{exact} (b) L_3U_2 (c) L_4U_2

(d) $L_3U_2 - L_2U_{exact}$ (e) $L_4U_2 - L_2U_{exact}$

Figure 2.5: Comparison of truncation error measures based on second order solution

exact truncation error can be calculated and used to compare the truncation error estimate. Figure 2.5 shows the results based on the second order solution. The top row is the second-order flux integral based on the exact solution, 2.5a, third-order flux integral based on second-order solution, 2.5b, and fourth-order flux integral based on second-order solution, 2.5c, respectively. The differences between the estimates of the truncation error and the exact truncation error are shown at the bottom row. Careful investigation of these two figures reveals that the differences are higher near the boundaries.

The same experiment has been done for third order solution and shown in Figure 2.6. The top row is the exact third order truncation error and an estimation of the truncation error calculated by the fourth order flux integral based on third order solution. Not surprisingly, the truncation error is less for this case; $O\,(1)$ for the third-order in comparison with $O\,(20)$ for the second order solution, and the difference between the error estimates, Figure 2.6c, is significantly smaller. However, similar to the previous



(a) L_3U_{exact} (b) L_4U_3

(c) $L_4U_3 - L_3U_{exact}$

Figure 2.6: Comparison of truncation error measures based on third order solution

case, the difference is larger near the boundaries.

Figure 2.7 illustrates the convergence of the L_2-norm of the truncation error with mesh refinement for different orders of exact truncation error and estimates of the truncation error obtained by p-TE approach for this Poisson problem. In calculating the flux integral for the Poisson problem, one order of accuracy is lost in calculating the flux (the gradients) and another order is lost in calculating the flux integral. Consequently, the exact second order truncation error shown by the solid black line is zero order and similarly for the exact third and fourth order truncation error which are first and second order, respectively. Not surprisingly, by increasing the order of accuracy, the magnitude of the truncation error decreases. Similarly, if the truncation error is estimated by a higher-order flux integral based on the second order solution, zero order convergence is obtained, as plotted by the dashed lines with squares, and first order convergence results from a fourth order flux integral based on the third order solution, as plotted by the dashed line with triangles.

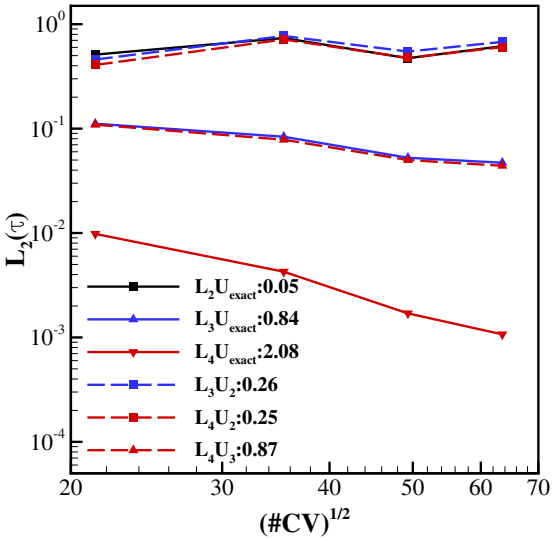


Figure 2.7: Truncation error behavior with mesh refinement for Poisson

2.2 Eigendecomposition of the Truncation Error

The discretization error, ε, can be directly related to the truncation error, τ, by the error transport equation [80]:

$$\frac{\partial R_h}{\partial U_h}\varepsilon = \tau \tag{2.10}$$

in which the truncation error estimated based on the numerically approximated solution serves as a source term and $\frac{\partial R_h}{\partial U_h}$ is the discrete Jacobian matrix. Both truncation and discretization errors can be expanded as a combination of right eigenvectors of the Jacobian, giving

$$\frac{\partial R_h}{\partial U_h}\sum_i b_i \boldsymbol{x_i} = \sum_i a_i \boldsymbol{x}_i \tag{2.11}$$

where a_i and b_i are the weights in the eigendecomposition of the truncation and discretization error, respectively. Arranging the left hand side, we get:

$$\sum_i b_i \frac{\partial R_h}{\partial U_h}\boldsymbol{x_i} = \sum_i a_i \boldsymbol{x}_i \tag{2.12}$$

and using the definition of eigenvalues, λ_i

$$\sum_i b_i \lambda_i \boldsymbol{x_i} = \sum_i a_i \boldsymbol{x}_i \tag{2.13}$$

Because the x_i are linearly independent, we must have

$$a_i = b_i \lambda_i \tag{2.14}$$

which links the eigendecomposition of the two forms of error. Since the discrete Jacobian is not symmetric, the right $\boldsymbol{x}_j$ and left eigenvectors $\boldsymbol{y}_i$[2] are different and with proper normalization, the two sets of eigenvectors are orthonormal:

$$\boldsymbol{y}_i \cdot \boldsymbol{x}_j = \delta_{ij} \tag{2.15}$$

This property can be used to evaluate the coefficients in the eigendecomposition of the truncation error:

$$\boldsymbol{y}_i \cdot \tau = \boldsymbol{y}_i \cdot \sum_i a_i \boldsymbol{x}_i = \sum_i a_i \boldsymbol{y}_i \cdot \boldsymbol{x}_i = \sum_i a_i \delta_{ij} = a_j \tag{2.16}$$

and consequently the discretization error coefficients by Equation 2.14. We have used eigenanalysis as a novel tool to relate the discretization and truncation error to each other and to compare the behavior of a wide range of different discretization schemes commonly used for diffusive flux approximation in a cell-centered unstructured finite volume solver [37]. Here, we focus on what eigenanalysis can tell us about truncation error estimation.

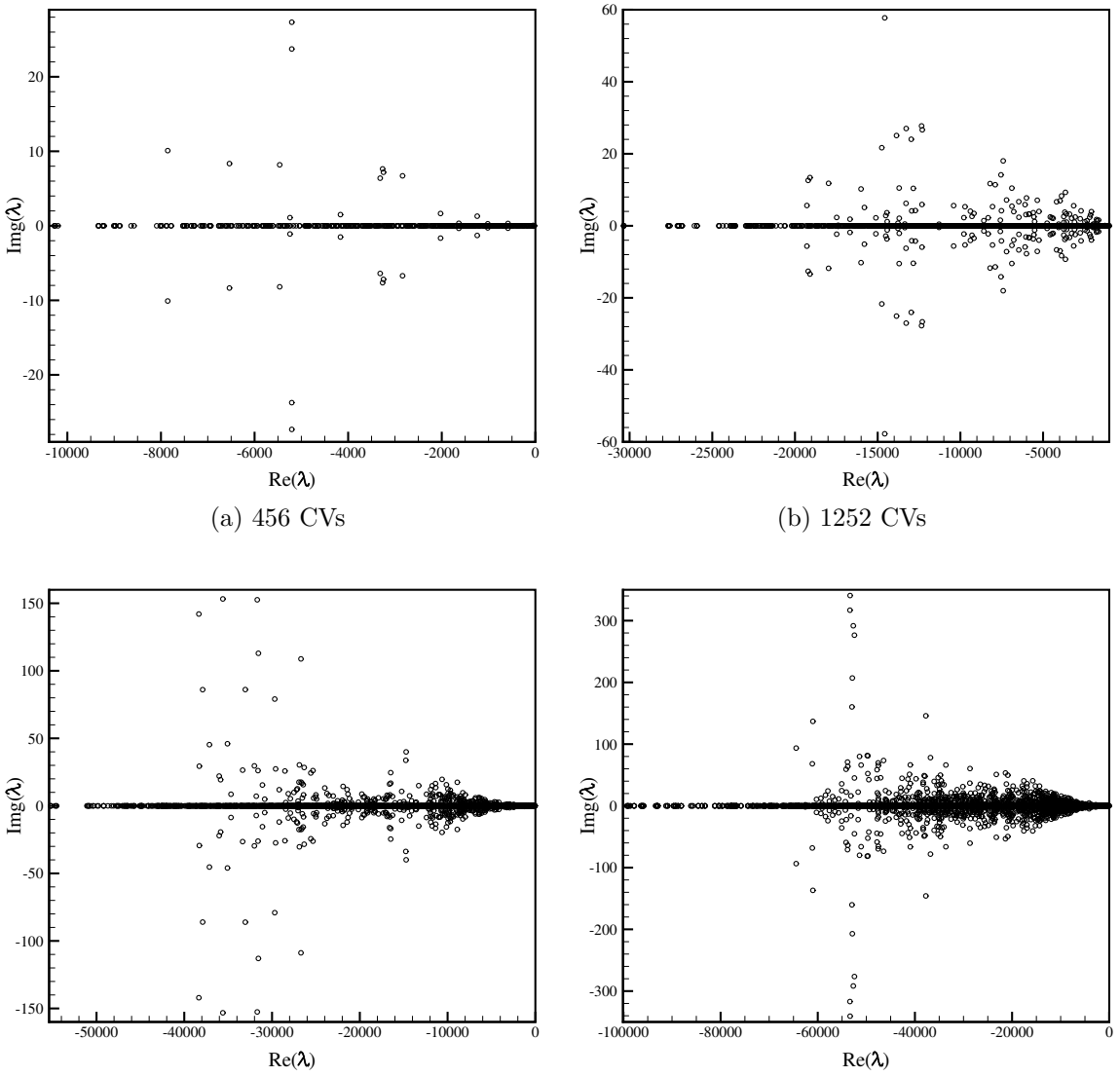
2.3 Rough Mode Dominance in Truncation Error

In this section, we will show that the rough modes dominate the truncation error for unstructured meshes and therefore, the p-TE estimation method is not capable of estimating the truncation error on unstructured mesh as accurately as on a structured mesh.

2.3.1 Poisson Equation

The complex eigenvalues for a second-order cell-centered discretization of the problem described above is depicted for four meshes in Figure 2.8. As the Jacobian matrix is asymmetric, the eigenvalues are complex, though the magnitude of the imaginary part

[2] $\boldsymbol{y}_i$, the i^{th} left eigenvector, is a row vector.



(a) 456 CVs

(b) 1252 CVs

(c) 2436 CVs

(d) 4040 CVs

Figure 2.8: Eigenvalue spectra of the Poisson problem

Mesh size	Max EV (rightmost EV)	Min EV (leftmost EV)
456	-19.728	-10275.74
1252	-19.730	-30352.41
2436	-19.739	-55207.65
4040	-19.738	-99286.94

Table 2.1: Extremal eigenvalues for the Poisson test case

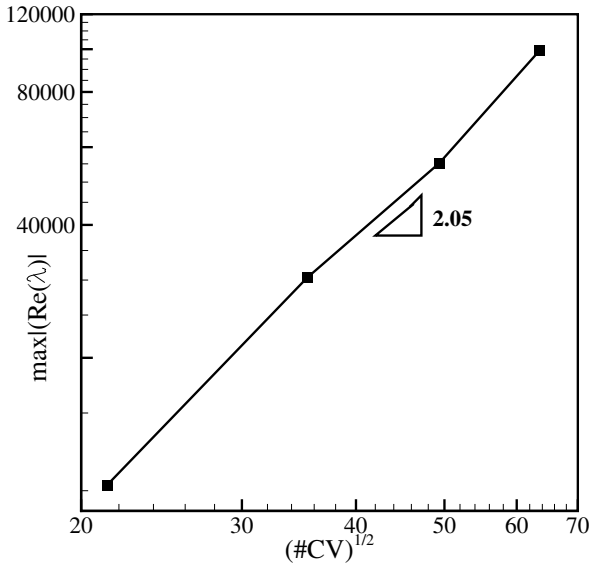
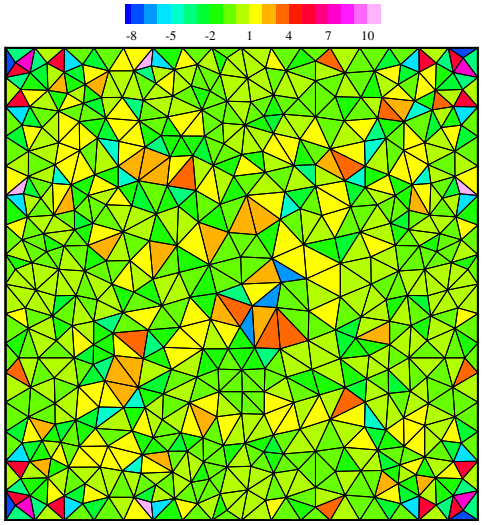


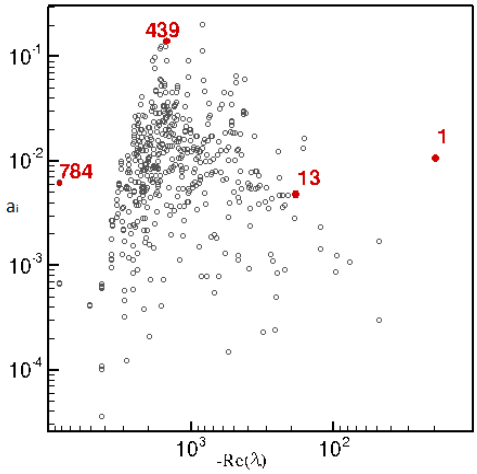
Figure 2.9: Largest eigenvalue versus mesh size for the Poisson problem

never exceeds about one percent of the magnitude of the real part. In all cases, there are a handful of eigenvalues scattered to the left of the rest of the distribution; because the eigenvectors associated with these eigenvalues are non-zero only in a handful of cells adjacent to the boundary, we expect these eigenvectors to be very sensitive to boundary condition implementation. The leftmost (largest) and the rightmost (smallest) eigenvalues for these meshes are shown in Table 2.1; by increasing the number of mesh points and consequently decreasing the cell size, the magnitude of leftmost eigenvalues increases while the rightmost eigenvalue is effectively zero regardless of the mesh size. Therefore, the leftmost eigenvalue scales linearly with the mesh size and is proportional to h^{-2} as shown in Figure 2.9 where the square root of the number of cells is considered as the mesh size.

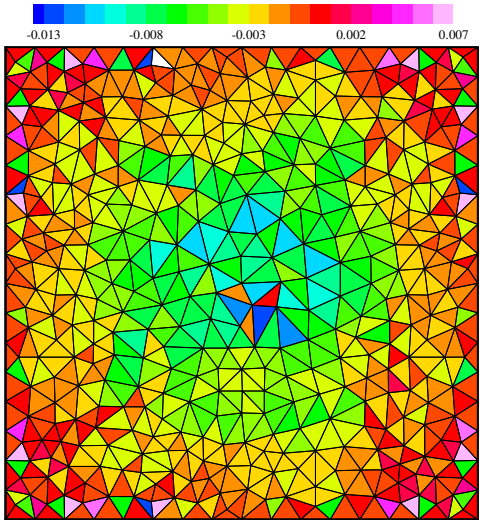
A full eigendecomposition of both truncation and discretization error is performed on the 1252 cell mesh and depicted in Figure 2.10. The magnitude of the weights in its decomposition is plotted against the real part of the eigenvalue, $Re\,(\lambda)$, and as indicated the rough modes dominate the unstructured mesh truncation error. To visualize the error weights more effectively, a logarithmic scale is used for both weights and real part of the eigenvalue. The scheme is stable and consequently $Re\,(\lambda)$ is negative and so the negative of the real part of the eigenvalue, $-Re\,(\lambda)$, is plotted instead. The truncation error is visually rough and this is supported by the magnitude of the the weights in its decomposition. The eigenvalues vary in magnitude from $O\,(1)$ to $O\,(h^{-2})$, with the



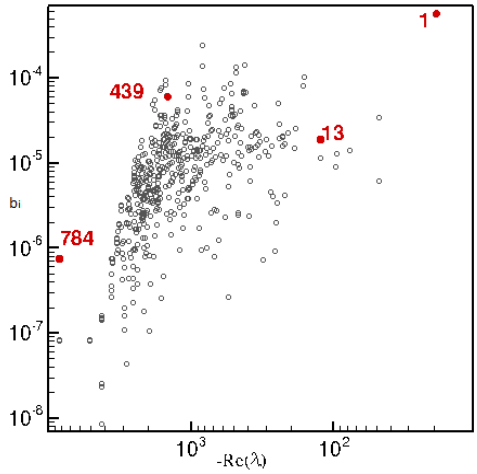
(a) Truncation error



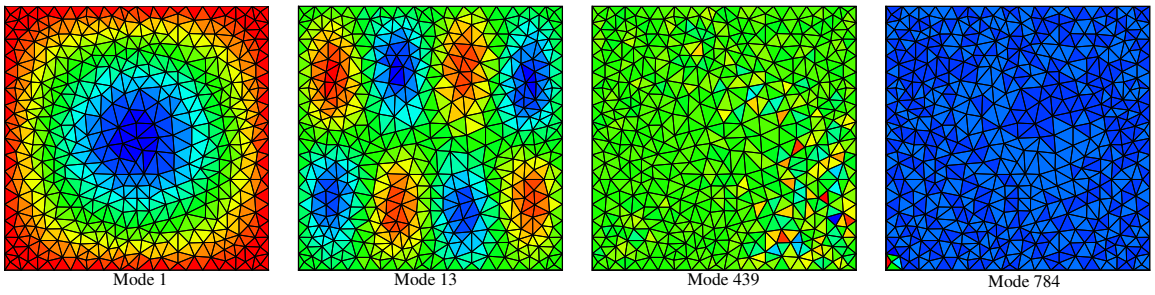
(b) Weights in decomposition of truncation error



(c) Discretization error



(d) Weights in decomposition of discretization error



(e) Mode shapes

Figure 2.10: Eigendecomposition of truncation and discretization error for Poisson

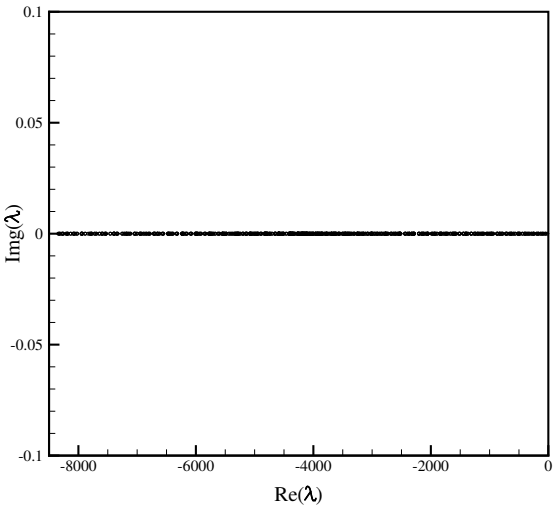
latter corresponding to rough modes. This implies that rough modes in the truncation error will be de-amplified by a factor of $O(h^2)$ as we solve the linear system, restoring the design order of accuracy for the discretization error and simultaneously making the discretization error much smaller than the truncation error. Investigation of this figure also reveals that the weight of the rough modes in the decomposition of the discretization error are much reduced. The largest weight in the eigendecomposition of the truncation error is 2×10^{-1}, while the corresponding weight for discretization error is much smaller and is close to 6×10^{-4} .

Four mode shapes are shown in Figure 2.10e. The smoothest mode corresponds to the $\sin(\pi x)\sin(\pi y)$ mode expected for homogeneous Dirichlet boundary conditions. The next mode shown is still a reasonable facsimile of a single Fourier mode. However, the next modes appear progressively more local and noisy.

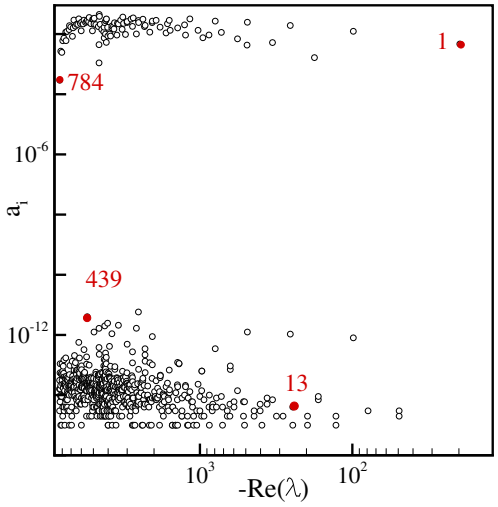
These rough modes in the truncation error are the consequences of using an unstructured mesh. If a perfect quadrilateral mesh is used instead, the Jacobian matrix is symmetric and consequently the imaginary part of the eigenvalue is zero for all modes as shown in Figure 2.11a. The number of cells for this perfect quadrilateral mesh is the same as the unstructured mesh shown in Figure 2.10. The weights in the eigendecomposition of truncation and discretization error are depicted in Figure 2.11b and 2.11c, respectively. The weights of the error terms for this perfect mesh are much smaller than the corresponding weights shown in Figure 2.10. This is expected since the perfect mesh provides more symmetry and smoothness. More smoothness is also proved by investigation of Figure 2.11d for which all highlighted modes in Figure 2.11b and 2.11c including the leftmost mode and the mode with largest weight have a smooth distribution. It should be noted that although the perfect mesh is used, the spatial discretization is for the unstructured mesh. Consequently, despite the fact that there is symmetry in the mesh, the full cancellation, similar to the structured mesh, is not obtained.

2.3.2 Euler Equation

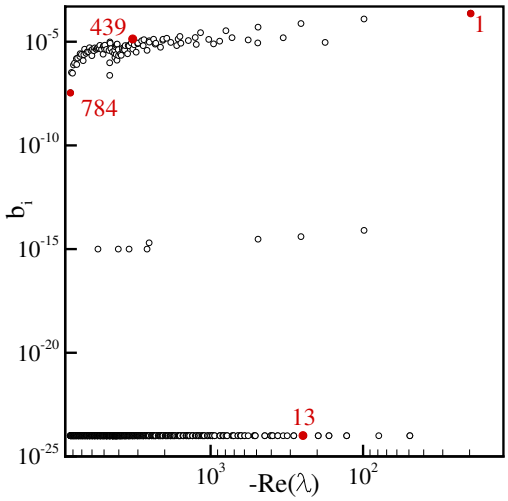
The supersonic vortex is an inviscid compressible (Euler) flow problem with an exact solution. Therefore, the exact truncation error can be calculated easily. The exact solution is:



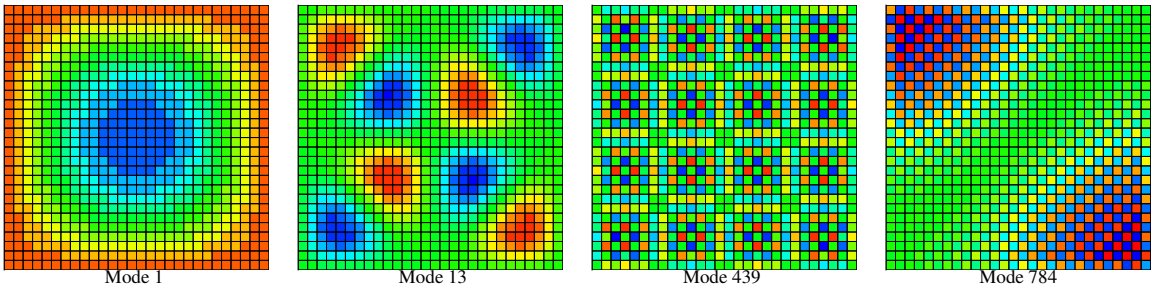
(a) Eigenvalue spectra



(b) Weights of truncation error

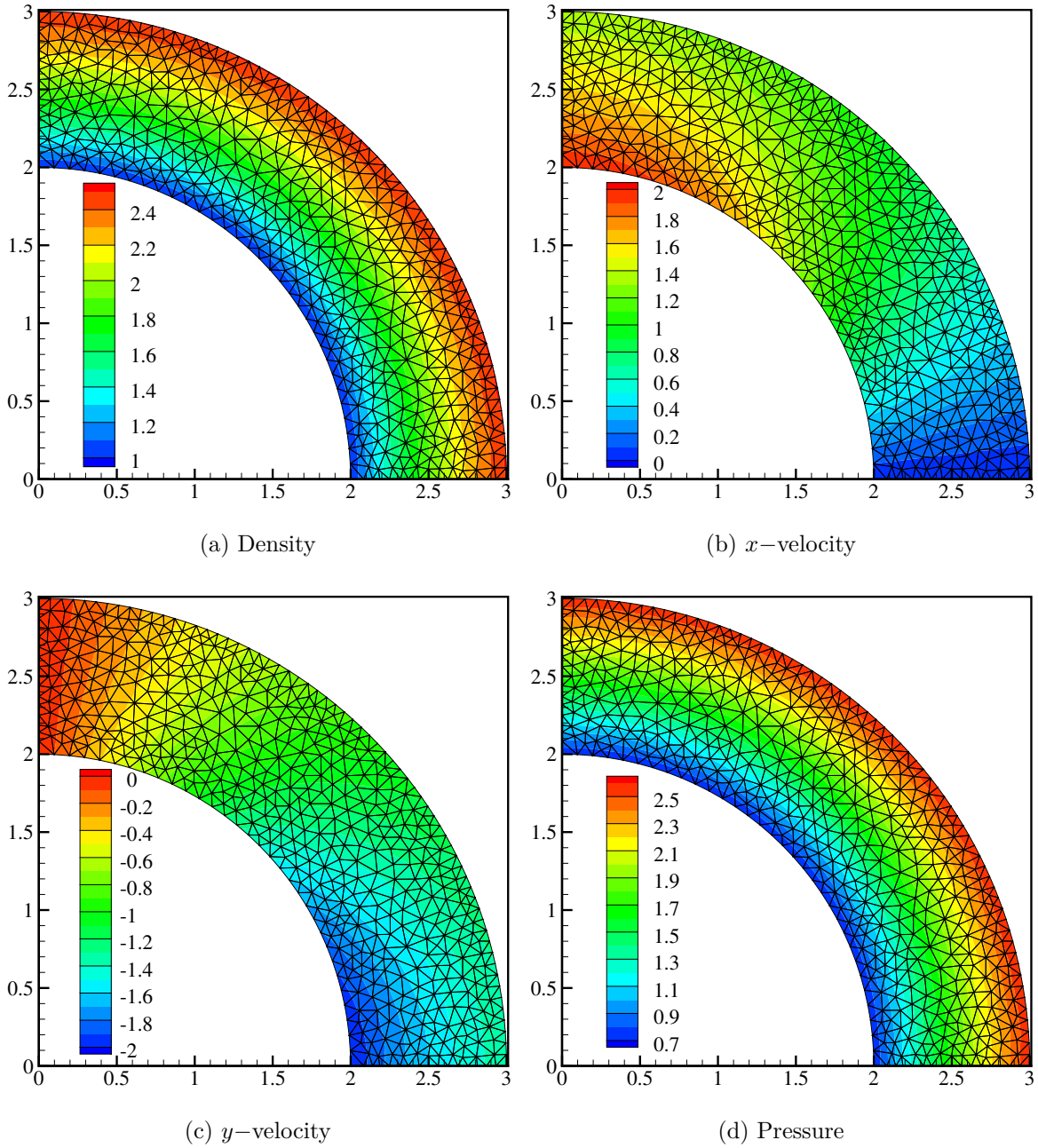


(c) Weights of discretization error



(d) Mode shapes

Figure 2.11: Eigendecomposition of truncation and discretization error for a perfect quadrilateral mesh for Poisson



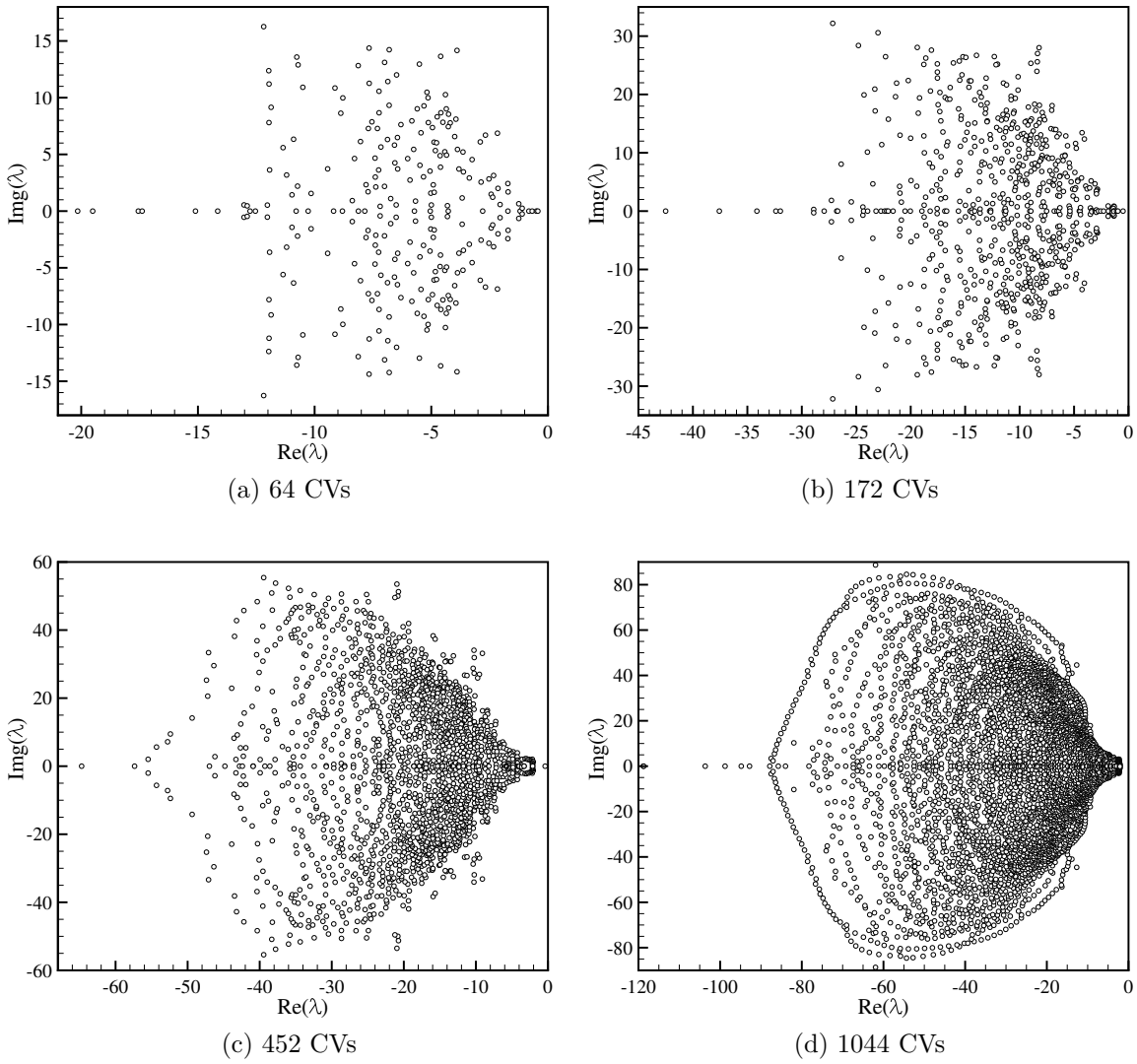
(a) Density

(b) $x-$velocity

(c) $y-$velocity

(d) Pressure

Figure 2.12: Exact solution of supersonic vortex



(a) 64 CVs

(b) 172 CVs

(c) 452 CVs

(d) 1044 CVs

Figure 2.13: Eigenvalue spectra, supersonic vortex

Mesh size	Max EV (rightmost EV)	Min EV (leftmost EV)
64	-0.431	-20.154
172	-0.502	-42.503
452	-0.391	-64.706
1044	-2.102	-98.787

Table 2.2: Extremal eigenvalues for the Euler test case

$$
\begin{aligned}
\rho &= \left(1 + \frac{\gamma - 1}{2}\left(1 - \frac{R_i^2}{R^2}\right) M_i^2\right)^{\frac{1}{\gamma - 1}} \\
u_x &= \frac{y R_i M_i}{R^2} \\
u_y &= \frac{-x R_i M_i}{R^2} \\
p &= \frac{\rho^\gamma}{\gamma}
\end{aligned}
$$

where we take $R_i = 2.0$ as the inner radius and $M_i = 2.0$ is the inlet Mach number. The four components of the solution for the supersonic vortex are depicted in Figure 2.12. The eigenvalue decomposition is shown in Figure 2.13 for four different meshes. By increasing the number of mesh points and consequently decreasing the cell size, the magnitude of the leftmost eigenvalues increases while the rightmost eigenvalue is effectively zero regardless of mesh size. Table 2.2 summarizes the maximum and minimum eigenvalues for these four meshes and the last column of the table illustrates that the leftmost eigenvalue scales with the square root of the mesh size, and therefore is proportional to h^{-1} as depicted in Figure 2.14. For the Poisson problem, Figure 2.8, the order of magnitude of the imaginary part of the eigenvalue is much smaller than the order of magnitude of the real part. Nevertheless, the magnitude of the imaginary part of the eigenvalue is comprable to the real part as shown Figure 2.13 and for several modes, the imaginary part of the eigenvalue is nonzero.

For the second finest mesh, eigendecomposition of truncation error and discretization error has been done and the results are shown in Figure 2.15 for the weights of these error terms in eigendecomposition. Similar to the Poisson test case, the weight of the truncation error is plotted versus the negative of the real part of the eigenvalue and the weights of the truncation error are larger than for the discretization error. The density component of the three highlighted modes are also depicted in Figure 2.15e. The left most mode is completely rough although a smooth distribution exists for the rightmost mode.

2.3.3 Discussion

As mentioned earlier, the truncation error is rough as a result of discontinuous jumps between control volumes in the coefficients of the terms in the Taylor series expansion of the error. Eigenanalysis and numerical experiments provide valuable insight into the behavior of truncation and discretization error for unstructured mesh discretization.

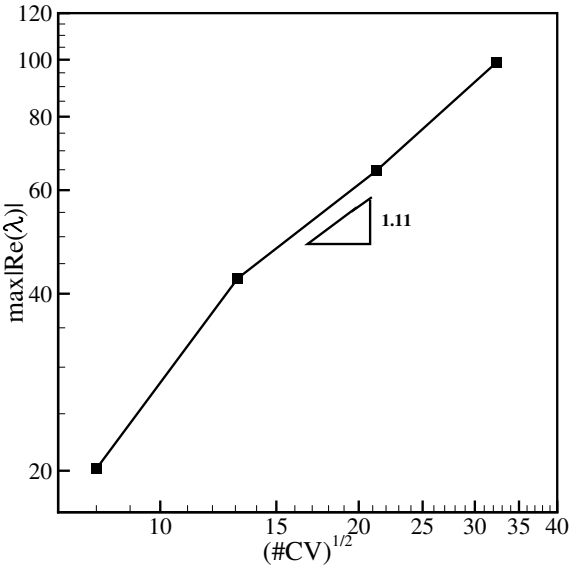
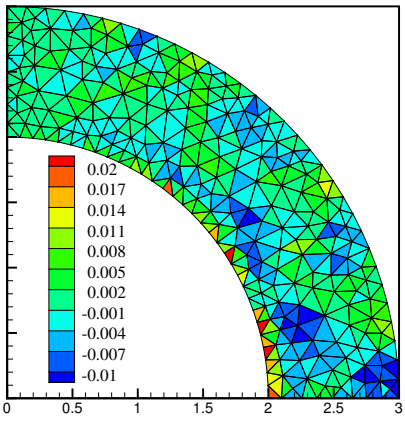
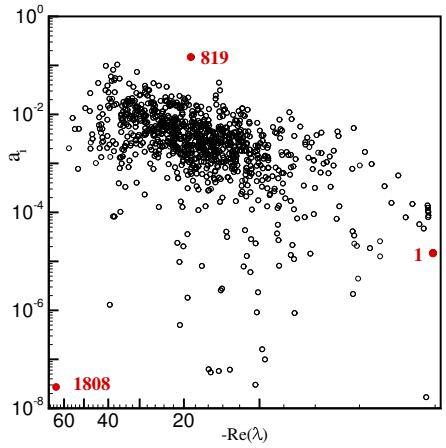


Figure 2.14: Largest eigenvalue versus mesh size for the Euler problem

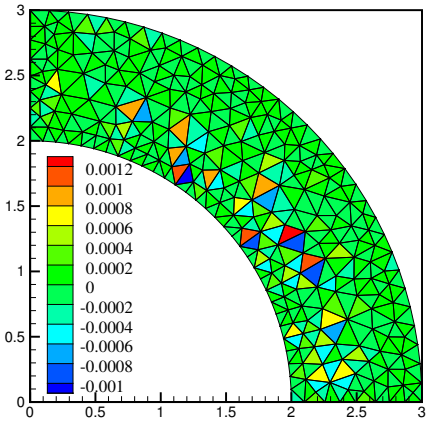
We have demonstrated that the differences in asymptotic behavior of the truncation and discretization error for unstructured meshes and the features of the truncation error can be predicted by the eigendecomposition of the discrete problem. The dominance of rough modes makes truncation error estimation more challenging when arbitrary unstructured meshes are used. If a perfectly symmetric mesh is used, the truncation error is much smoother and easier to estimate accurately even in high frequency modes. Nevertheless, it is not possible to generate perfect symmetric meshes for real application cases. Therefore, the dominance of rough error modes in the unstructured mesh truncation error suggests the need to develop a smooth truncation error estimate that can be computed using quantities already available in a typical finite volume flow solver.



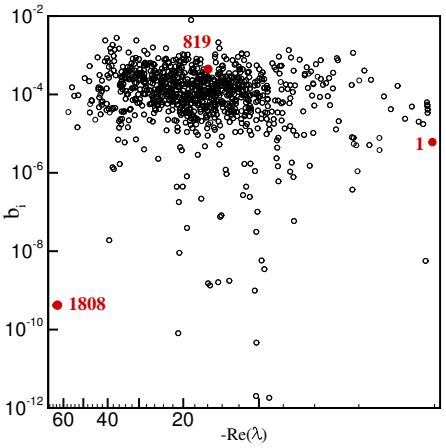
(a) Density component of truncation error



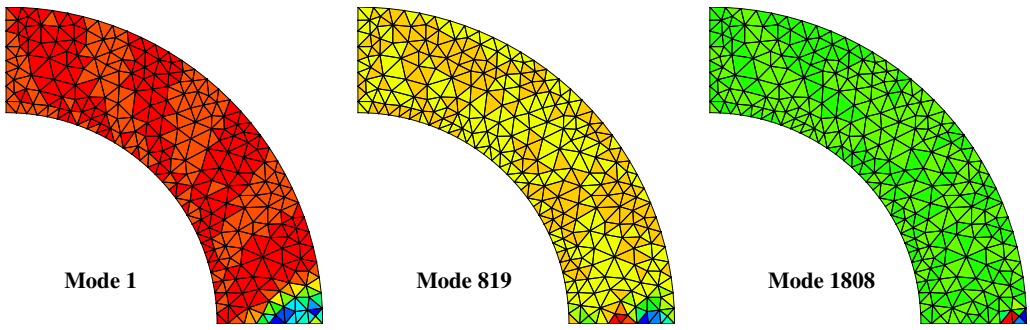
(b) Weights in decomposition of truncation error



(c) Density component of discretization error



(d) Weights in decomposition of discretization error



(e) Mode Shapes

Figure 2.15: Eigendecomposition of truncation and discretization error for Euler

Chapter 3

Truncation Error based on Interpolated Solution

In the previous chapter, we have shown that the rough modes are dominant in unstructured mesh truncation error. As a consequence , the simple p-truncation error estimate is not accurate. We do not expect to be able to use this p-truncation error estimate to improve the solution through defect correction or to estimate output functional error. In fact, we will show in the next chapters that the lack of smoothness of the truncation error is responsible for the poor performance of defect correction, output error estimation and mesh adaptation. Therefore, developing a smooth truncation error estimate is essential.

It has been discussed that the non-smooth distribution of the truncation error is the result of discontinuous jumps between control volumes. Using p-TE for estimating the truncation error, the higher-order flux integral is calculated based on the lower order solution for which the numerical fluxes are integrated over each control volume edges. The flux vector is calculated by solution reconstruction for each control volume at two sides of each edge separately, as shown in Figure 3.1, and therefore two different values are obtained for each quadrature point. By using an interpolation to approximate the solution data, we can find a single solution at each quadrature point and consequently, a smoother distribution of the truncation error may be obtained. Computing the truncation error using this continuous data is explained in Section 3.1. We have explored two approaches for solution interpolation: a simple spline interpolation (discussed in Section 3.2) and a more accurate C^1 interpolation of the solution (described in Section 3.3). Both of these interpolating schemes require vertex based data, but the solution we have is a cell-centered finite volume-based solution for which control volume averages are known. Accordingly, we need to convert the control-volume averaged data to point-wise data at vertices as shown schematically in Figure 3.2. We will reconstruct the control-volume data to the vertices of each cell and by averaging the solution from all cells incident on a vertex, we can find a single solution, or first and second deriva-

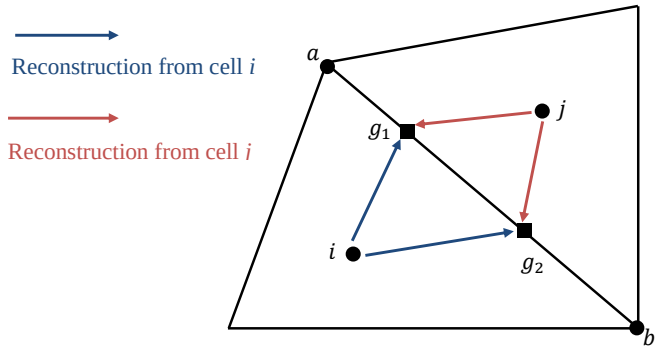


Figure 3.1: Two different fluxes at quadrature points

tives, at each vertex. We can then use the point-wise date to produce a continuous approximation of the solution. For the spline interpolation method, knowing the solution at vertices is enough. However, for C^1 interpolation, the solution and first and second derivatives are required. The pseudo-code for doing this conversion is shown in Algorithm 3.1 in which $sol[vert]$ is the solution or derivatives at each vertex.

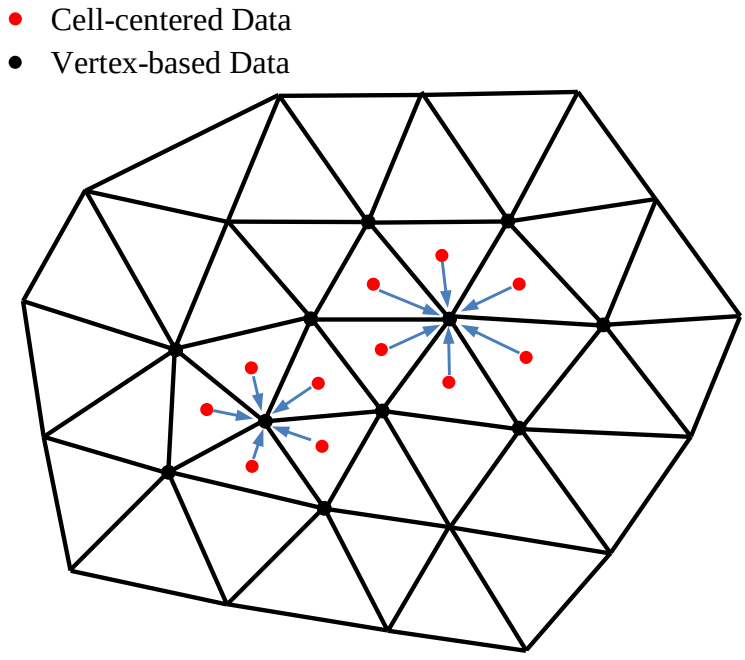


Figure 3.2: Conversion of the solution from cell-centered finite-volume based to vertex-centered

Algorithm 3.1 Conversion algorithm from cell-centered finite-volume based to vertex-based

```
for all cells
    for all three vertices
        sol[Vert]+=reconstructed solution at the vertex

for all vertices
    sol[Vert]/=number of cells incident on the vertex
```

3.1 Truncation Error Computation Using the Continuous Solution

Once a single value is obtained for the solution at each quadrature point, the corresponding flux vector can be calculated using this continuous solution. Instead of using a combination of the solution obtained from two neighboring cells, such as averaging, adding a jump term or the Roe's scheme, a single value for each quadrature point is obtained. To compute the flux integral for each control volume, numerical fluxes calculated based on the continuous data should be integrated over the control volume edges. The integration is done by the Gauss quadrature integration rule described in Chapter 1. As described, the truncation error estimate is the higher order flux integral based on the lower-order solution. As an example, consider using third-order flux integration based on the second-order solution to estimate the truncation error. To find the truncation error estimate based in the continuous data, we need to follow these steps.

1. The spline interpolation or the C^1 interpolation of the second-order converged solution should be obtained from the reconstructed data.

2. The continuous data is used to find the solution at two quadrature points on each edge.

3. The analytic flux vector is calculated based on the solution data on the quadrature points.

4. The higher order flux integral is calculated by integrating the flux over the control volume edges with the proper weight for each quadrature point as

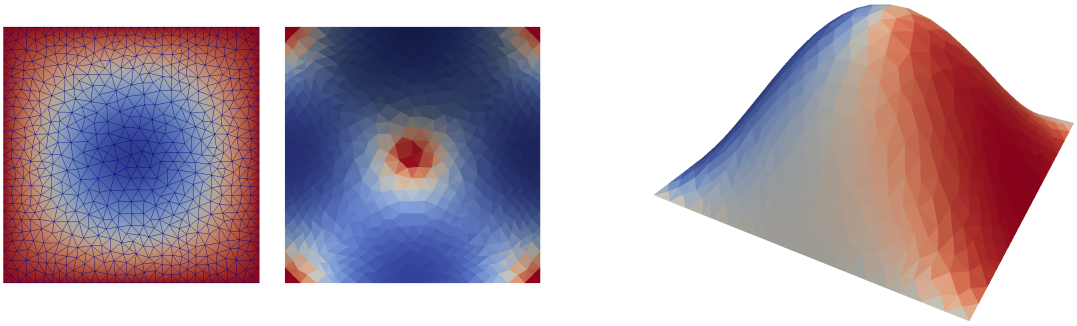
$$-\frac{1}{A_{CV_i}}\oint_{CS_i}\vec{F}\cdot\hat{n}\,ds=-\frac{1}{A_{CV_i}}\oint_{CS_i}\left(\left(\vec{F}_{g_1}\cdot\hat{n}\right)W_{g_1}+\left(\vec{F}_{g_2}\cdot\hat{n}\right)W_{g_2}\right)ds \tag{3.1}$$

where the coordinates of the quadrature points are

$$\begin{aligned} g_1 &= \frac{\mathbf{x}_b+\mathbf{x}_a}{2}+\frac{\mathbf{x}_b-\mathbf{x}_a}{2\sqrt{3}} \\ g_2 &= \frac{\mathbf{x}_b+\mathbf{x}_a}{2}-\frac{\mathbf{x}_b-\mathbf{x}_a}{2\sqrt{3}} \\ \hat{n} &= \frac{(y_a-y_b,x_b-x_a)}{|\mathbf{x}_a-\mathbf{x}_b|} \end{aligned} \tag{3.2}$$

3.2 Spline Interpolation using the Common Geometry Module (CGM)

The Common Geometry Module (CGM) is a code library providing geometry functionality for mesh generation. CGM includes a facet-based modeler which constructs a spline approximation to a three-dimensional surface triangulation [89, 90]. To generate this 3D surface, we provide a surface for which the x and y coordinates are the same as the coordinates of the original 2D mesh, and the z component is the numerical solution at that vertex. CGM produces a smooth cubic 3D surface spline from this data. Us-



(a) solution (left) and the CGM input surface (right) colored by the $z-$normal

(b) 3D view of the CGM surface colored by the $x-$normal

Figure 3.3: CGM's input surface

ing this smooth surface, a unique solution value is obtained at each quadrature point because there is a surface fitted to the solution as depicted in Figure 3.3. It should be noted that the splines are generated based on the coordinates of the mesh and the solution at vertices. Accordingly, there is no guarantee that a point with exactly the same x and y components of the quadrature points exists on the 3D surface. This implies that it is not possible to find the continuous solution of a quadrature points just by specifying its coordinates and there is not a specific value for any arbitrary $z\,(x, y)$. We find the projection of the point with the x and y coordinates of the quadrature point on the 3D surface and use the x and y coordinates of the projected point and continue the process until the coordinates of the projected point are close to the quadrature point within a specified tolerance as shown in Figure 3.4. The blue line represents the 3D surface, the black point shows the coordinates of the quadrature point and the red point is the projected point on the 3D surface. The z coordinate of the projected point is the single value solution at the quadrature point calculated from two neighboring cells. As finding the diffusive flux requires the gradients, we use the functions available in this code library to find the normal on each facet as shown on Figure 3.3b and the x and y components of the normal vector is proportional to the derivative of the solution with respect to x and y, respectively or the gradients in these directions.

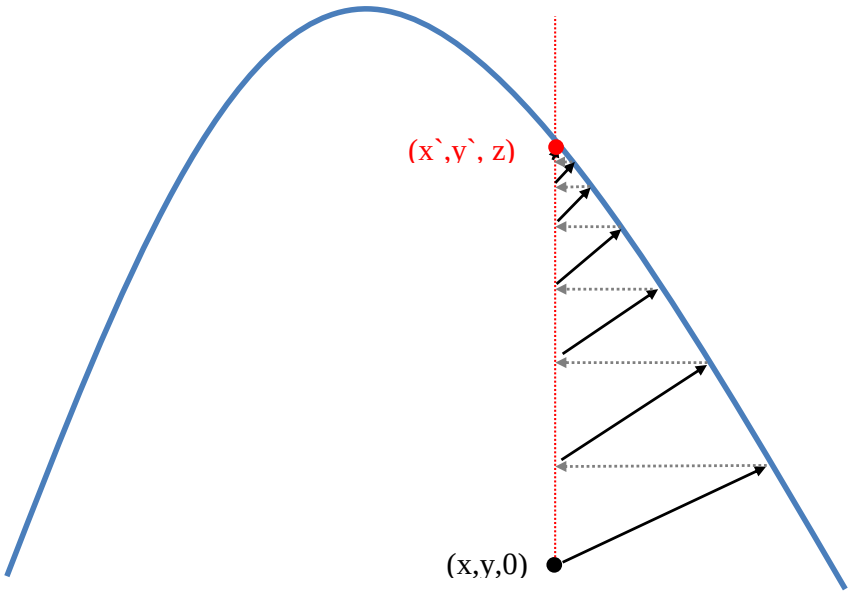


Figure 3.4: Finding the solution on quadrature points by projection on the 3D spline interpolation using CGM

We have generated a fourth order spline based on exact solutions at vertices for which the expected accuracy of the solution and gradients are fourth order and third order, respectively. Figure 3.5 illustrates the order of accuracy of the solution and gradient for the Poisson test case described in the previous chapter. As a manufactured solution is set, the solution and gradients at any arbitrary point including quadrature points can be calculated easily and in this figure, the L_2-norm of the error is plotted versus the mesh size. The solution at quadrature points is fourth order accurate as expected, but the gradient calculation is only second-order. Inaccurate normal computation is the consequence of assigning a single normal per facet as shown in Figure 3.6 where two neighboring cells sharing the quadrature point have different normal vectors. Therefore, computing gradients based on this normal calculation is inaccurate.

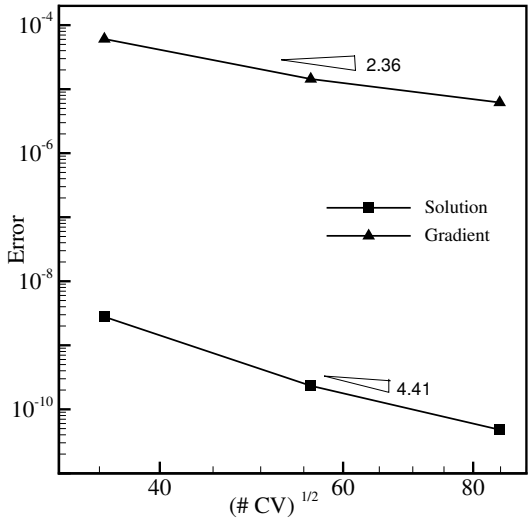


Figure 3.5: The normal and solution calculation using CGM

The interpolated values of the solution are used to calculate the flux integral for each control volume as an estimate of the truncation error. The comparison between the exact second-order truncation error, the original estimate of the truncation error (based on the non-smoothed discrete solution) and the modified estimate of the truncation error (based on the smoothed interpolated solution) is shown on the top row of Figure 3.7. Detailed investigation of the differences between the exact and estimated truncation error on the bottom row shows that the modified estimation of the truncation error is smoother than the original one and is closer to the exact value. It should be noted that we do not expect to obtain a smooth distribution of the truncation error, similar to

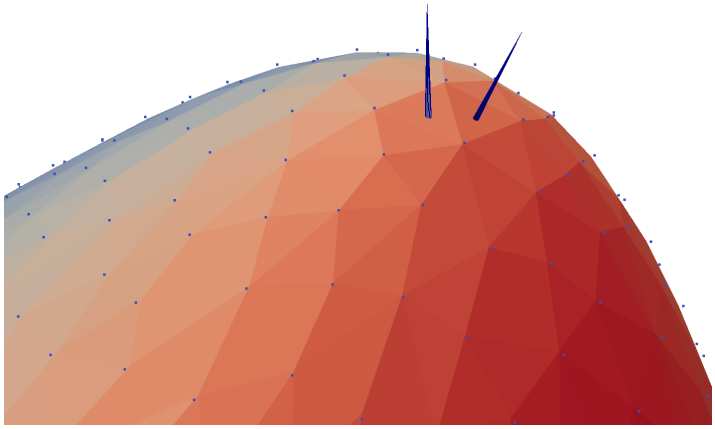
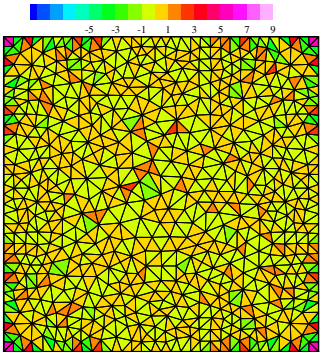
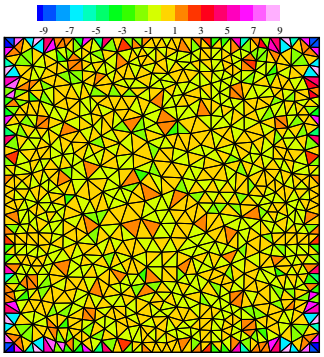


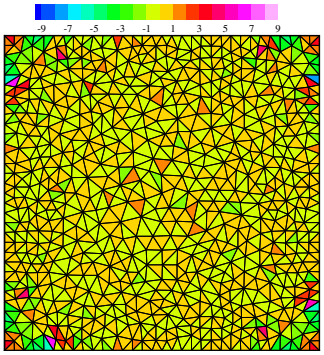
Figure 3.6: Inaccurate normal calculation using CGM



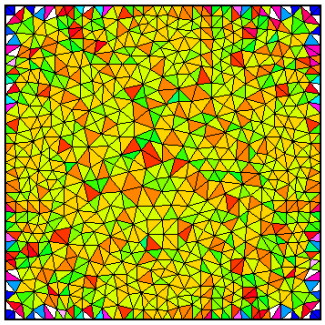
(a) L_2U_{exact}



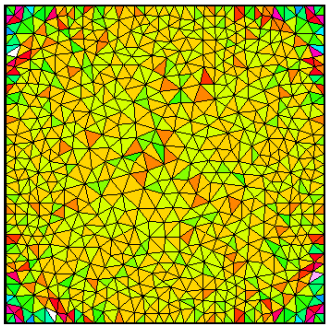
(b) $L_3^{non-smooth}U_2$



(c) $L_3^{smooth,CGM}U_2$



(d) L_2U_{exact}-$L_3^{non-smooth}U_2$



(e) $L_2U_{exact}-L_3^{smooth,CGM}U_2$

Figure 3.7: Comparison of truncation error based on interpolated solution by CGM

the truncation error on structured meshes, and the only goal of interpolating is to find a single value for solutions and gradients at quadrature points. As the exact solution is known, the exact truncation error can be calculated easily. The error in calculating the truncation error is shown for three different meshes in Figure 3.8 for the original estimate of the truncation error using the discrete solution and the interpolated solution by CGM. The error using the interpolated solution is less than the discrete solution.

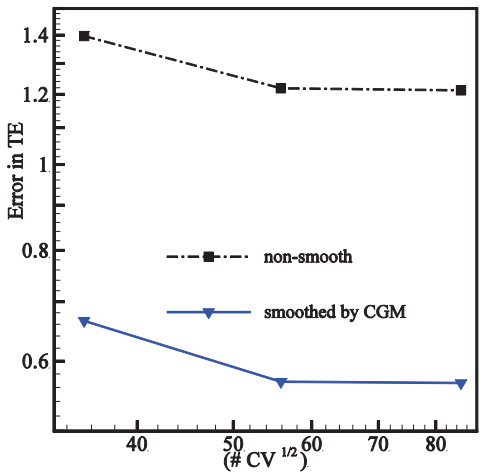


Figure 3.8: Error in calculating the truncation error using the discrete solution and interpolated solution by CGM

Generally, inspite of having a unique solution value, a single value for gradients is not obtained if CGM is used as the interpolating technique. Accordingly, a more accurate interpolated scheme which gives smoothed solution and gradients is needed.

3.3 C^1 Interpolation of the Solution

To overcome the gradient problem with CGM's spline interpolation, we have chosen to compute a C^1 interpolation that approximates the reconstructed finite volume solution and then we use this interpolation to evaluate the analytic fluxes and integrate them to compute a truncation error estimate.

To compute the C^1 interpolation, we use the Argyris element because of its high rate of spatial convergence and simplicity of implementation compared to other elements. The Argyris triangle [7, 15], which is a form of Hermite interpolation, is based on the fifth order space of quintic polynomials over a triangle. This element produces C^2 continuity at the vertices and full C^1 continuity along the edges between elements of a

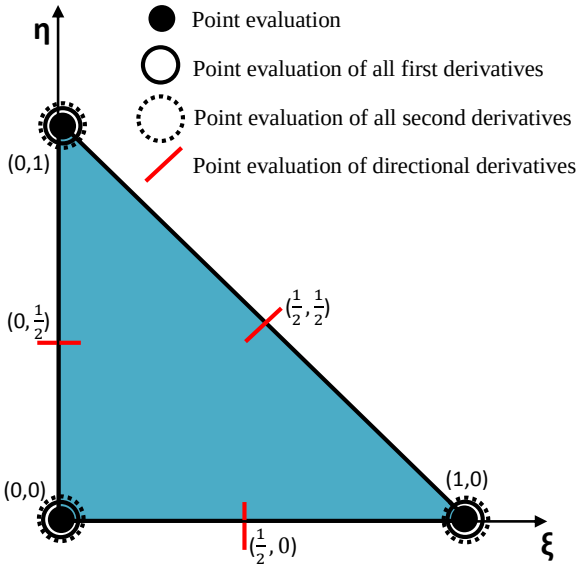


Figure 3.9: Argyris reference element

triangulation, which suffices for our case as it is able to provide us the same flux vector based on the solution and gradients in two neighboring cells.

Quintic polynomials in $\mathbb{R}^2$ are a 21-dimensional space; the Argyris triangle specifies these 21 degrees of freedom by using six pieces of data per vertex and one per edge. The vertex degrees of freedom are the solution, two first derivatives to specify the gradient, and three second derivatives to specify the unique components of the (symmetric) Hessian matrix; the edge degree of freedom is the normal derivative as shown in Figure 3.9. To compute these values, we first reconstruct the finite volume solution to higher (third or fourth) order. For each edge or vertex, the reconstruction gives multiple values for the solution and its derivatives, one per cell. For a p^{th} order reconstruction, these solution values will differ only by $O\left(h^p\right)$ for smooth data, with derivatives losing one order each. We take an arithmetic average of these values as our input to the interpolation

as described at the beginning of this chapter. The solution is then represented as

$$\begin{aligned}
u \;=\;& u_{(0,0)}\varphi_0 + \left.\frac{\partial u}{\partial \xi}\right|_{(0,0)}\varphi_1 + \left.\frac{\partial u}{\partial \eta}\right|_{(0,0)}\varphi_2 + \left.\frac{\partial^2 u}{\partial \xi^2}\right|_{(0,0)}\varphi_3 + \left.\frac{\partial^2 u}{\partial \xi \partial \eta}\right|_{(0,0)}\varphi_4 + \left.\frac{\partial^2 u}{\partial \eta^2}\right|_{(0,0)}\varphi_5 \\
& + u_{(1,0)}\varphi_6 + \left.\frac{\partial u}{\partial \xi}\right|_{(1,0)}\varphi_7 + \left.\frac{\partial u}{\partial \eta}\right|_{(1,0)}\varphi_8 + \left.\frac{\partial^2 u}{\partial \xi^2}\right|_{(1,0)}\varphi_9 + \left.\frac{\partial^2 u}{\partial \xi \partial \eta}\right|_{(1,0)}\varphi_{10} + \left.\frac{\partial^2 u}{\partial \eta^2}\right|_{(1,0)}\varphi_{11} \\
& + u_{(0,1)}\varphi_{12} + \left.\frac{\partial u}{\partial \xi}\right|_{(0,1)}\varphi_{13} + \left.\frac{\partial u}{\partial \eta}\right|_{(0,1)}\varphi_{14} + \left.\frac{\partial^2 u}{\partial \xi^2}\right|_{(0,1)}\varphi_{15} + \left.\frac{\partial^2 u}{\partial \xi \partial \eta}\right|_{(0,1)}\varphi_{16} + \left.\frac{\partial^2 u}{\partial \eta^2}\right|_{(0,1)}\varphi_{17} \\
& + \left.\left(\frac{\partial u}{\partial \xi}n_\xi + \frac{\partial u}{\partial \eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)}\varphi_{18} \\
& + \left.\left(\frac{\partial u}{\partial \xi}n_\xi + \frac{\partial u}{\partial \eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)}\varphi_{19} \\
& + \left.\left(\frac{\partial u}{\partial \xi}n_\xi + \frac{\partial u}{\partial \eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)}\varphi_{20}
\end{aligned} \tag{3.3}$$

where the φ_i are the 21 basis functions for the Argyris element. Each of these has a value of one for its own variable and zero for all others. For instance, the derivative of φ_7 with respect to ξ is nonzero at point $(1,0)$, $\left.\frac{\partial \varphi_1}{\partial \xi}\right|_{(1,0)} = 1$, and all other derivatives are zero at that point. The magnitude and all derivatives of φ_1 are zero at the other points. Each basis function is a quintic polynomial:

$$\begin{aligned}
\varphi_i(\xi,\eta) \;=\;& c_{0,0} \\
+\;& c_{1,0}\xi + c_{0,1}\eta \\
+\;& c_{2,0}\xi^2 + c_{1,1}\xi\eta + c_{0,2}\eta^2 \\
+\;& c_{3,0}\xi^3 + c_{2,1}\xi^2\eta + c_{1,2}\xi\eta^2 + c_{0,3}\eta^3 \\
+\;& c_{4,0}\xi^4 + c_{3,1}\xi^3\eta + c_{2,2}\xi^2\eta^2 + c_{1,2}\xi\eta^3 + c_{0,4}\eta^4 \\
+\;& c_{5,0}\xi^5 + c_{4,1}\xi^4\eta + c_{3,2}\xi^3\eta^2 + c_{2,3}\xi^2\eta^3 + c_{1,4}\xi\eta^4 + c_{0,5}\eta^5
\end{aligned} \tag{3.4}$$

each of which has exactly 21 degrees of freedom. To determine these basis functions, these 21 constants should be evaluated based on non-zero constraints. The non-zero constraint on the basis functions are:

$$
\begin{array}{lllllllll}
\varphi_0\big|_{(0,0)} & = & 1 & , & \varphi_6\big|_{(1,0)} & = & 1 & , & \varphi_{12}\big|_{(0,1)} & = & 1 \\
\frac{\partial\varphi_1}{\partial\xi}\Big|_{(0,0)} & = & 1 & , & \frac{\partial\varphi_7}{\partial\xi}\Big|_{(1,0)} & = & 1 & , & \frac{\partial\varphi_{13}}{\partial\xi}\Big|_{(0,1)} & = & 1 \\
\frac{\partial\varphi_2}{\partial\eta}\Big|_{(0,0)} & = & 1 & , & \frac{\partial\varphi_8}{\partial\eta}\Big|_{(1,0)} & = & 1 & , & \frac{\partial\varphi_{14}}{\partial\eta}\Big|_{(0,1)} & = & 1 \\
\frac{\partial^2\varphi_3}{\partial\xi^2}\Big|_{(0,0)} & = & 1 & , & \frac{\partial^2\varphi_9}{\partial\xi^2}\Big|_{(1,0)} & = & 1 & , & \frac{\partial^2\varphi_{15}}{\partial\xi^2}\Big|_{(0,1)} & = & 1 \\
\frac{\partial^2\varphi_4}{\partial\xi\partial\eta}\Big|_{(0,0)} & = & 1 & , & \frac{\partial^2\varphi_{10}}{\partial\xi\partial\eta}\Big|_{(1,0)} & = & 1 & , & \frac{\partial^2\varphi_{16}}{\partial\xi\partial\eta}\Big|_{(0,1)} & = & 1 \\
\frac{\partial^2\varphi_5}{\partial\eta^2}\Big|_{(0,0)} & = & 1 & , & \frac{\partial^2\varphi_{11}}{\partial\eta^2}\Big|_{(1,0)} & = & 1 & , & \frac{\partial^2\varphi_{17}}{\partial\eta^2}\Big|_{(0,1)} & = & 1
\end{array}
$$

$$
\begin{array}{lcccl}
\left(\frac{\partial\varphi_{18}}{\partial\xi}n_\xi + \frac{\partial\varphi_{18}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} & = & \frac{\partial\varphi_{18}}{\partial\eta}\Big|_{\left(\frac{1}{2},0\right)} & = & 1 \\
\left(\frac{\partial\varphi_{19}}{\partial\xi}n_\xi + \frac{\partial\varphi_{19}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} & = & -\frac{\sqrt{2}}{2}\left(\frac{\partial\varphi_{19}}{\partial\xi} + \frac{\partial\varphi_{19}}{\partial\eta}\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} & = & 1 \\
\left(\frac{\partial\varphi_{20}}{\partial\xi}n_\xi + \frac{\partial\varphi_{20}}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} & = & -\frac{\partial\varphi_{20}}{\partial\xi}\Big|_{\left(0,\frac{1}{2}\right)} & = & 1
\end{array}
$$

All elements are mapped into a unique Argyris element and the solution and derivatives are computed in the reference element and mapped back to the physical element. Figure 3.10 shows the linear transformation from the original element to the reference element. The linear transformation is of the form:

$$
\begin{aligned}
x &= a_0 + a_1\xi + a_2\eta \\
y &= b_0 + b_1\xi + b_2\eta
\end{aligned}
\tag{3.5}
$$

where according to the coordinates of the reference element:

$$
\left\{\begin{array}{l} x_1 = a_0 \\ y_1 = b_0 \end{array}\right. , \left\{\begin{array}{l} x_2 = a_0 + a_1 \\ y_2 = b_0 + b_1 \end{array}\right. , \left\{\begin{array}{l} x_3 = a_0 + a_2 \\ y_3 = b_0 + b_2 \end{array}\right.
\tag{3.6}
$$

Substituting in Eq. 3.5, the linear transformation constant is obtained as:

$$
\begin{aligned}
x &= x_1 + (x_2 - x_1)\,\xi + (x_3 - x_1)\,\eta \\
y &= y_1 + (y_2 - y_1)\,\xi + (y_3 - y_1)\,\eta
\end{aligned}
\tag{3.7}
$$

Or ξ and η in terms of x and y as:

$$
\begin{aligned}
\xi &= \frac{x\,(y_3 - y_1) + y\,(x_1 - x_3) + y_1x_3 - x_1y_3}{(y_3 - y_1)\,(x_2 - x_1) - (x_3 - x_1)\,(y_2 - y_1)} \\
\eta &= \frac{x\,(y_2 - y_1) + y\,(x_1 - x_2) + y_1x_2 - x_1y_2}{(y_2 - y_1)\,(x_3 - x_1) - (x_2 - x_1)\,(y_3 - y_1)}
\end{aligned}
\tag{3.8}
$$

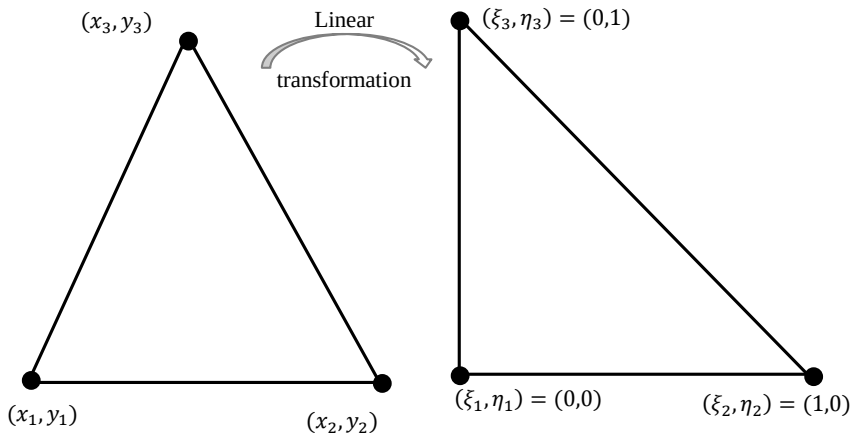


Figure 3.10: Linear mapping from physical to Argyris reference element

According to Eq. 3.3, the solution is a function of the derivatives of the solution with respect to the coordinates of the reference element. To be able to calculate everything in the reference space, we must write these derivatives in terms of the derivatives in the physical space, which are easy to calculate by reconstruction, and the geometric transformation:

$$\begin{aligned} \frac{\partial u}{\partial \xi} &= \frac{\partial u}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial \xi} = \frac{\partial u}{\partial x}(x_2 - x_1) + \frac{\partial u}{\partial y}(y_2 - y_1) \\ \frac{\partial u}{\partial \eta} &= \frac{\partial u}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial \eta} = \frac{\partial u}{\partial x}(x_3 - x_1) + \frac{\partial u}{\partial y}(y_3 - y_1) \end{aligned} \tag{3.9}$$

In addition to the first derivatives, the second derivatives are required to obtain the C^1

interpolation of the solution according to Eq. 3.3. Second derivatives are calculated as:

$$\begin{aligned}\frac{\partial^2 u}{\partial \xi^2} &= \frac{\partial}{\partial \xi}\left(\frac{\partial u}{\partial x}\frac{\partial x}{\partial \xi}+\frac{\partial u}{\partial y}\frac{\partial y}{\partial \xi}\right)=\frac{\partial^2 x}{\partial \xi^2}\frac{\partial u}{\partial x}+\frac{\partial^2 y}{\partial \xi^2}\frac{\partial u}{\partial y}\\ &+ \left(\frac{\partial x}{\partial \xi}\right)^2\frac{\partial^2 u}{\partial x^2}+2\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \xi}\frac{\partial^2 u}{\partial x\partial y}+\left(\frac{\partial y}{\partial \xi}\right)^2\frac{\partial^2 u}{\partial y^2}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2 u}{\partial \xi\partial \eta} &= \frac{\partial}{\partial \xi}\left(\frac{\partial u}{\partial x}\frac{\partial x}{\partial \eta}+\frac{\partial u}{\partial y}\frac{\partial y}{\partial \eta}\right)=\frac{\partial^2 x}{\partial \xi\partial \eta}\frac{\partial u}{\partial x}+\frac{\partial^2 y}{\partial \xi\partial \eta}\frac{\partial u}{\partial y}\\ &+ \frac{\partial x}{\partial \xi}\frac{\partial x}{\partial \eta}\frac{\partial^2 u}{\partial x^2}+\left(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta}+\frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta}\right)\frac{\partial^2 u}{\partial x\partial y}+\frac{\partial y}{\partial \xi}\frac{\partial y}{\partial \eta}\frac{\partial^2 u}{\partial y^2}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2 u}{\partial \eta^2} &= \frac{\partial}{\partial \eta}\left(\frac{\partial u}{\partial x}\frac{\partial x}{\partial \eta}+\frac{\partial u}{\partial y}\frac{\partial y}{\partial \eta}\right)=\frac{\partial^2 x}{\partial \eta^2}\frac{\partial u}{\partial x}+\frac{\partial^2 y}{\partial \eta^2}\frac{\partial u}{\partial y}\\ &+ \left(\frac{\partial x}{\partial \eta}\right)^2\frac{\partial^2 u}{\partial x^2}+2\frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \eta}\frac{\partial^2 u}{\partial x\partial y}+\left(\frac{\partial y}{\partial \eta}\right)^2\frac{\partial^2 u}{\partial y^2}\end{aligned} \tag{3.10}$$

which are summarized as:

$$\begin{aligned}\frac{\partial^2 u}{\partial \xi^2} &= (x_2-x_1)^2\frac{\partial^2 u}{\partial x^2}+2(x_2-x_1)(y_2-y_1)\frac{\partial^2 u}{\partial x\partial y}+(y_2-y_1)^2\frac{\partial^2 u}{\partial y^2}\\ \frac{\partial^2 u}{\partial \xi\partial \eta} &= (x_2-x_1)(x_3-x_1)\frac{\partial^2 u}{\partial x^2}+(y_2-y_1)(y_3-y_1)\frac{\partial^2 u}{\partial y^2}\\ &+((x_3-x_1)(y_2-y_1)+(x_2-x_1)(y_3-y_1))\frac{\partial^2 u}{\partial x\partial y}\\ \frac{\partial^2 u}{\partial \eta^2} &= (x_3-x_1)^2\frac{\partial^2 u}{\partial x^2}+2(x_3-x_1)(y_3-y_1)\frac{\partial^2 u}{\partial x\partial y}+(y_3-y_1)^2\frac{\partial^2 u}{\partial y^2}.\end{aligned} \tag{3.11}$$

Accordingly, all terms in Eq. 3.3 are known and the interpolated solution and derivatives can be calculated easily. The final point that needs to be considered here is the normal derivative directions. It was mentioned that there are six piece of data for each vertex and the other three are from the normal derivatives at the center of each edge. The normal derivative directions are shown in Figure 3.11. The constants that we obtained for the basis functions using Eq. 3.3 are based on these normal directions.

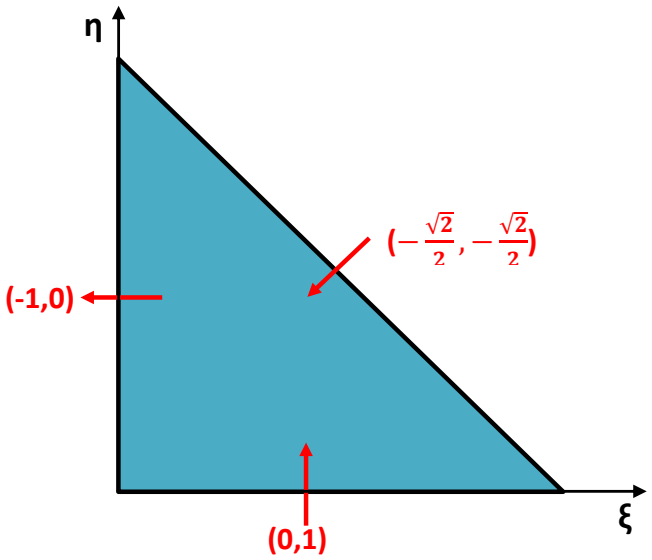


Figure 3.11: Normal derivative directions for C^1 interpolation of the solution

As an example, consider the nonzero equation corresponding to φ_6 :

$$\begin{aligned}
\varphi_6|_{(1,0)} &= c_{0,0} + c_{1,0}\xi + c_{0,1}\eta + c_{2,0}\xi^2 + c_{1,1}\xi\eta + c_{0,2}\eta^2 + c_{3,0}\xi^3 + c_{2,1}\xi^2\eta \\
&+ c_{1,2}\xi\eta^2 + c_{0,3}\eta^3 + c_{4,0}\xi^4 + c_{3,1}\xi^3\eta + c_{2,2}\xi^2\eta^2 + c_{1,2}\xi\eta^3 + c_{0,4}\eta^4 \\
&+ c_{5,0}\xi^5 + c_{4,1}\xi^4\eta + c_{3,2}\xi^3\eta^2 + c_{2,3}\xi^2\eta^3 + c_{1,4}\xi\eta^4 + c_{0,5}\eta^5 \\
&= c_{0,0} + c_{1,0} + c_{2,0} + c_{3,0} + c_{4,0} + c_{5,0} = 1
\end{aligned}$$

and all other equations corresponding to φ_6 are zero, including the the basis function at other two vertices:

$$\varphi_6|_{(0,0)} = \varphi_6|_{(1,1)} = 0$$

or all derivatives of φ_6 at all three vertices:

$$\begin{aligned}
\left.\frac{\partial\varphi_6}{\partial\xi}\right|_{(1,0)} &= c_{1,0} + 2c_{2,0}\xi + c_{1,1}\eta + 3c_{3,0}\xi^2 + 2c_{2,1}\xi\eta + c_{1,2}\eta^2 + 4c_{4,0}\xi^3 + 3c_{3,1}\xi^2\eta \\
&+ 3c_{2,2}\xi\eta^2 + c_{1,2}\eta^3 + 5c_{5,0}\xi^4 + 4c_{4,1}\xi^3\eta + 3c_{3,2}\xi^2\eta^2 + 2c_{2,3}\xi\eta^3 + c_{1,4}\eta^4 \\
&= c_{1,0} + 2c_{2,0}\xi + 3c_{3,0} + 4c_{4,0} + 5c_{5,0} = 0
\end{aligned}$$

and the normal derivatives at all three edge centers:

$$\left(\frac{\partial\varphi_6}{\partial\xi}n_\xi + \frac{\partial\varphi_6}{\partial\eta}n_\eta\right)\Bigg|_{\left(\frac{1}{2},\frac{1}{2}\right)} = -\frac{\sqrt{2}}{2}\left(\frac{\partial\varphi_6}{\partial\xi} + \frac{\partial\varphi_6}{\partial\eta}\right) = 0$$

As a result, there are 21 equations corresponding to each basis function from each one is nonzero and the other 20 equations are zero. The full list of constraint equations is shown in Appendix A. These 21 equations are set for 21 basis functions and a (21×21) system should be solved to obtain the $c_{i,j}$ coefficients. This system of (21×21) equations is independent of the solution data and the corresponding coefficients for the basis functions are as listed in Table 3.1. It should be noted that this system of equations is solved only once and all elements are mapped into a single Argyris reference element and accordingly this post-processing step is cheap in terms of time and memory. A comparison between the CPU time required for second, third and fourth-order solutions and finding the C^1 interpolation of the solution is shown in Table 3.2 for two problems described in the previous chapter. Investigation of Table 3.2 shows that the post processing time required for C^1 interpolation of the solution is small compared to the solution time, and scales linearly with problem size, whereas the solution time grows faster than linearly. The CPU time required for C^1 interpolation increases with increasing reconstruction order, because the cost of reconstructing the solution and first and second derivatives of the solution increases.

	$c_{0,0}$	$c_{1,0}$	$c_{0,1}$	$c_{2,0}$	$c_{1,1}$	$c_{0,2}$	$c_{3,0}$	$c_{2,1}$	$c_{1,2}$	$c_{0,3}$	$c_{4,0}$	$c_{3,1}$	$c_{2,2}$	$c_{1,3}$	$c_{0,4}$	$c_{5,0}$	$c_{4,1}$	$c_{3,2}$	$c_{2,3}$	$c_{1,4}$	$c_{5,0}$
φ_0	1						-10			-10	15		-30		15	-6		30	30		-6
φ_1		1					-6		-11		8		10	18		-3		1	-10	-8	
φ_2			1					-11		-6		18	10		8		-8	-10	1		-3
φ_3				0.5			-1.5				1.5		-1.5			-0.5		1.5	1		
φ_4					1			-4	-4			5	10	5			-2	-6	-6	-2	
φ_5						0.5				-1.5			-1.5		1.5			1	1.5		-0.5
φ_6							10				-15		15			6		-15	-15		
φ_7							-4				7		-3.5			-3		3.5	3.5		
φ_8								-5				14	18.5				-8	-18.5	-13.5		
φ_9							0.5				-1		0.25			0.5		-0.25	-0.25		
φ_{10}								1				-3	-3.5				2	3.5	2.5		
φ_{11}													1.25					-0.75	-1.25		
φ_{12}										10			15		-15			-15	-15		6
φ_{13}									-5				18.5	14				-13.5	-18.5	-8	
φ_{14}										-4			-3.5		7			3.5	3.5		-3
φ_{15}													1.25					-1.25	-0.75		
φ_{16}									1				-3.5	-3				2.5	3.5	2	
φ_{17}										0.5			0.25		-1			-0.25	-0.25		0.5
φ_{18}								16				-32	-32				16	32	16		
φ_{19}													$8\sqrt{2}$					$-8\sqrt{2}$	$-8\sqrt{2}$		
φ_{20}									-16				32	32				-16	-32	-16	

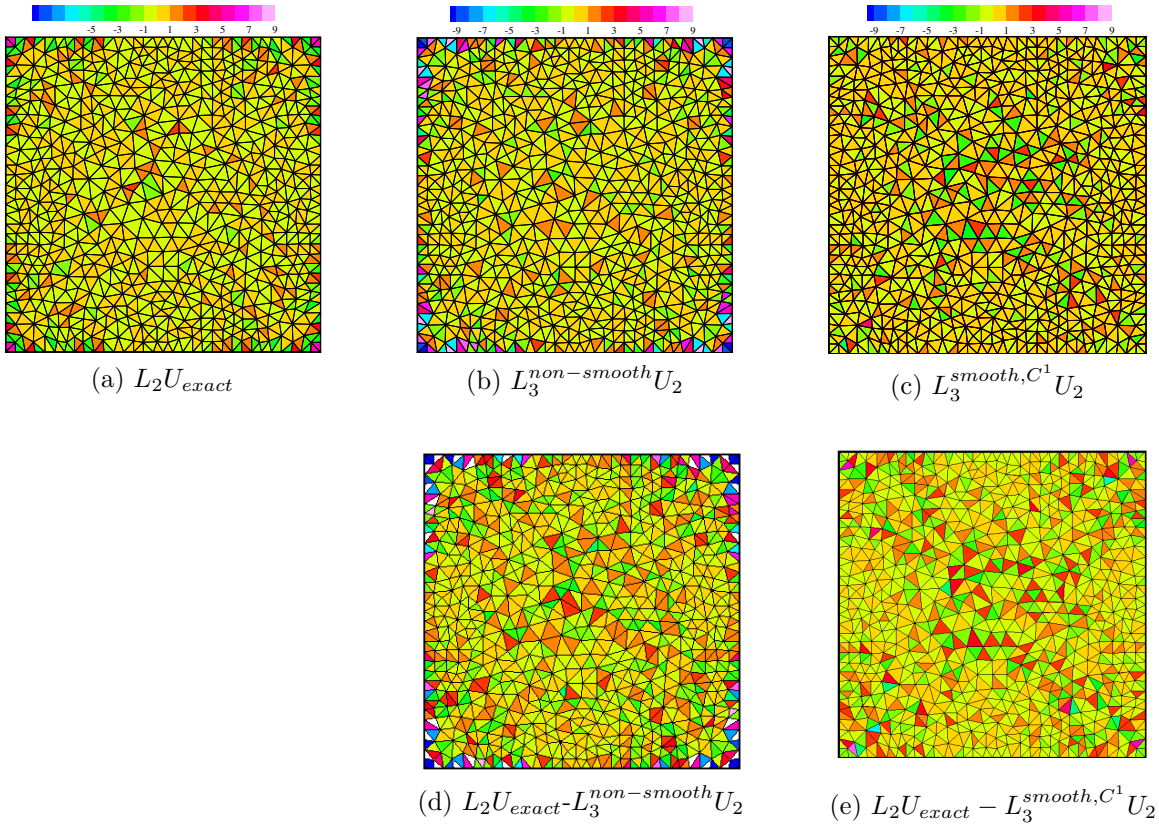
Table 3.1: Non-zero coefficients in Eq. 3.3 for the Argyris reference element

	$2^{nd} order$	$3^{rd} order$		$4^{th} order$	
$\#CVs$	$solution$	$solution$	$C^1\, interp$	$solution$	$C^1\, interp$
			$Poisson$		
1260	0.143	0.353	0.069	0.615	0.201
5004	0.795	1.635	0.306	2.635	0.852
19862	9.305	14.418	1.257	18.495	3.251
80430	76.875	130.210	5.092	158.069	13.192
			$Euler$		
1044	0.197	0.448	0.063	0.843	0.171
4408	0.959	2.072	0.291	3.915	0.723
16256	7.787	10.993	1.111	18.658	2.718
65502	94.411	202.422	4.446	264.728	11.626

Table 3.2: CPU time (sec) comparison for solution and C^1 interpolation of the solution

The interpolated values of the solution are used to calculate the flux integral for each control volume as an estimate of the truncation error. The comparison between the original estimate of the truncation error (based on the non-smoothed discrete solution) and the modified estimate of the truncation error (based on the smoothed solution by C^1 interpolation of the solution) is shown on the top row of Figure 3.12. The comparison of the difference between the exact truncation error and the estimate of the truncation error is shown on the bottom row and careful investigation shows that the smoothed estimation of the truncation error is smoother than the non-smoothed one and is closer to the exact value.

The same error plot is shown in Figure 3.13 for the original estimate of the truncation error using the discrete solution and the C^1 interpolation of the solution. The error using the interpolated solution is less than the discrete solution.



(a) L_2U_{exact} (b) $L_3^{non-smooth}U_2$ (c) $L_3^{smooth,C^1}U_2$

(d) L_2U_{exact}-$L_3^{non-smooth}U_2$ (e) $L_2U_{exact} - L_3^{smooth,C^1}U_2$

Figure 3.12: Comparison of truncation error based on C^1 interpolation of the solution

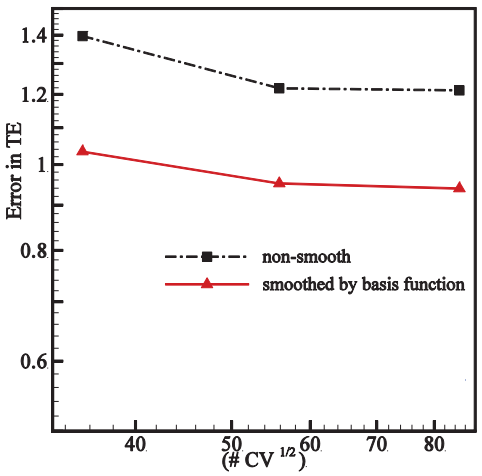


Figure 3.13: Error in calculating the truncation error using the discrete solution and C^1 interpolation of the solution

Chapter 4

Defect Correction

The history of defect correction in CFD goes back to the work by Pereyra [69]. The goal is to iteratively correct the original numerical solution so that it converges to the exact solution, Stetter [87] generalized this work. Kurzen et al. [46] used a global spline fit to generate one-dimensional and two-dimensional functions based on a numerical solution. The global spline fit was treated as a known exact solution and a second discrete solution was computed with appropriate source terms (similar to the method of manufactured solutions) to compute the error in the nearby problem. A slightly different method of defect correction was used by Naumovich et al. [55] which was a discrete approach. Instead of finding or creating a nearby solution to estimate the discretization error, the numerical solution is passed into a higher order discretization scheme to compute a residual. A first-order numerical solution is computed and defect correction is used to correct the solution to second-order accuracy. The motivation is that first-order problems are more robust and have better conditioned linear systems. The result is a quicker, more robust second-order accurate solution compared with solving the second-order problem directly.

4.1 General Algorithm

The basis of defect correction discretization error estimation relies on adding the truncation error as a source term which removes the local source of error related to the discretization of the domain and governing equations.

To perform defect correction, we should first solve the continuous problem with the source term f:

$$L\left(U\right) = f \tag{4.1}$$

discretely at order p to get $U_{h,p}$. The discretization error of the discrete solution converges at order p with mesh refinement. If the exact truncation error is added to

the source term of the problem:

$$\underbrace{L\left(U\right) = f}_{Continuous} + \underbrace{L_p\left(U_{h,exact}\right)}_{Discrete} \tag{4.2}$$

we can solve the modified discrete problem at p order to obtain a defect corrected solution. In theory, this defect corrected solution is the exact solution and therefore, the exact solution is obtained by solving the problem at p order which can be as cheap as solving the problem at second order. Note that this requires the exact truncation error.

Instead of using the exact p^{th} order truncation error, an estimation of the truncation error may be used. As mentioned earlier, the higher order flux integral based on lower order solution is an estimation of the truncation error which can be added to the source term of the original problem:

$$\underbrace{L\left(U\right) = f}_{Continuous} + \underbrace{L_{p+j}\left(U_{h,p}\right)}_{Discrete} \tag{4.3}$$

We can solve the modified discrete problem at lower order and the discretization error of this new defect corrected problem is reduced at order $p + j$ with mesh refinement instead of p^{th} order which means that higher order convergence is obtained by solving the problem at lower order. This is the behavior of defect correction on structured meshes. On unstructured meshes, on the other hand, the accuracy of the truncation error estimate may be an issue as described in Chapter 2.

Defect correction methods are extremely simple to implement as they only require the formulation of the truncation error estimate and the ability to include a source term in the discrete solver. In addition, defect correction methods are generally much less costly to solve than the original discrete system as they can be initialized using the already available discrete solution.

4.2 Defect Correction Based on Exact Truncation Error

Consider the Poisson problem in a (1×1) square

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 2\pi^2 \sin(\pi x) \sin(\pi y)$$

with homogeneous Dirichlet boundary condition for which the exact solution is:

$$U_{exact} = -\sin(\pi x) \sin(\pi y)$$

Since the exact solution is known, the exact truncation error can be computed and added to the source term to correct the solution as depicted in Figure 4.1. The dashed lines are the original second, third and fourth order solution which are shown in black, blue and red, respectively. The second order solution converges quadratically and, the fourth order solution converges quartically as expected, but the third order solution

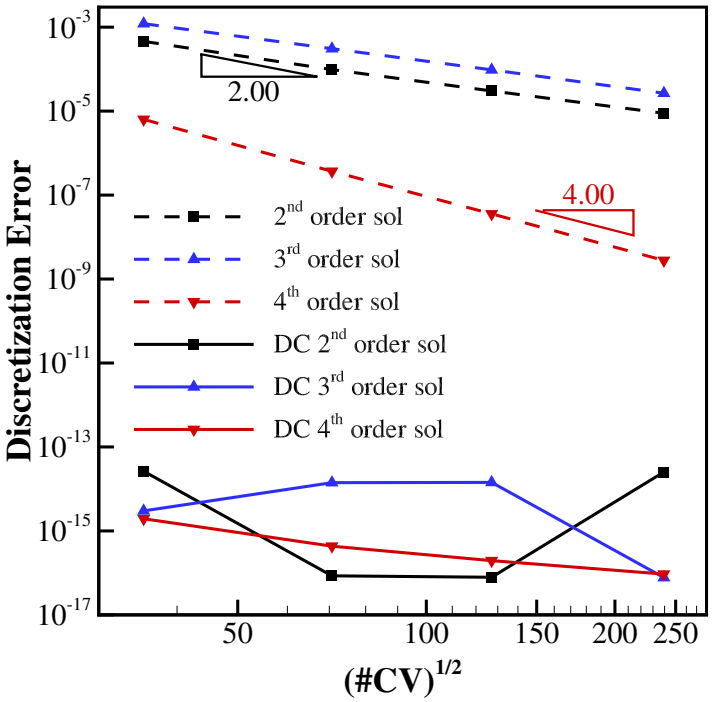


Figure 4.1: Defect correction based on exact truncation error for Poisson

converges quadratically and not cubically. This is the typical behavior of diffusive fluxes for third-order finite volume unstructured schemes. The solid lines are the defect corrected solutions based on adding the exact truncation error of the same order to the source term. The discretization error of all three corrected solutions are at the order of machine zero, implying that the exact solution is obtained. Therefore, the theory of defect correction works properly for the Poisson problem if the exact truncation error is added to the source term.

Having studied the diffusive fluxes using the Poisson equation, the performance of defect correction based on the exact solution is examined for convective fluxes. For this purpose, the advection equation with zero source term is considered. The velocity vector is set in x direction as $(1, 0)$, and the inlet velocity is set as $\sin(\pi y)$, so the problem and its exact solution are

$$\frac{\partial u}{\partial x} = 0 \Rightarrow u_{exact} = \sin(\pi y)$$

The problem domain considered is a (5×1) rectangular channel and the exact solution is shown in Figure 4.2. Similar to the Poisson problem, the convergence results are shown in Figure 4.3. The dashed lines are the original solutions and the solid lines are defect corrected solution based on exact solution. In contrast to the Poisson case, the third order solution converges cubically with mesh refinement. The discretization error of the corrected solutions are again at the order of machine zero, implying that the exact solution is obtained. Accordingly, the theory of defect correction works properly for the advection problem if the exact truncation error is added to the source term.

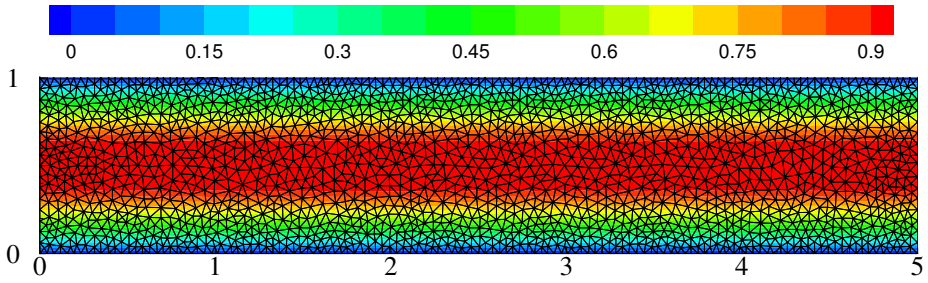


Figure 4.2: Exact solution of the advection problem

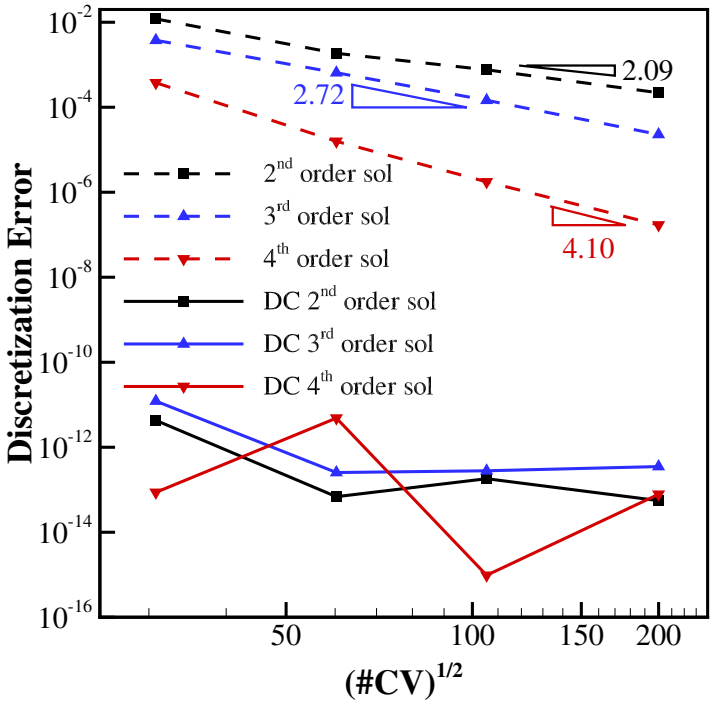


Figure 4.3: Defect correction based on exact truncation error for advection

4.3 Defect Correction for a Perfect Mesh

Instead of adding the exact truncation error to the source term for defect correction, an estimation of truncation error based on a higher order flux integral may be used. For this purpose, a perfect mesh with equilateral triangle elements is considered. The exact solution is set such that a zero boundary condition is obtained:

$$U_{exact} = -\sin(\pi y)\sin\left(\pi\left(y-\sqrt{3}x\right)\right)\sin\left(\pi\left(y+\sqrt{3}x-\sqrt{3}\right)\right) \tag{4.4}$$

Figure 4.4 shows the solution distribution in this equilateral triangular domain. Calculating the corresponding source term for this problem is easy.

The discretization error of the original second order problem and the defect corrected solution calculated by third and fourth order flux integrals based on the second order solution is depicted in Figure 4.5. The discretization error of the corrected solution by the exact truncation error is approximately zero as expected, but the corresponding error of the corrected solution by p-TE method is larger and comparable to the original problem. Not surprisingly, if $L_4(U_2)$ is added to the source term, the discretization

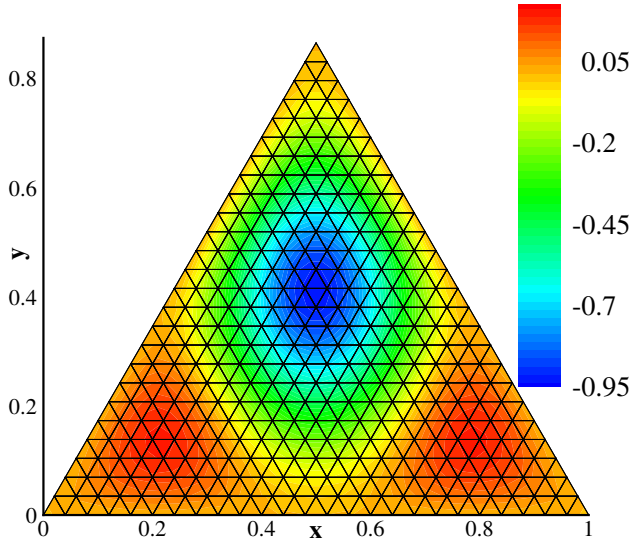
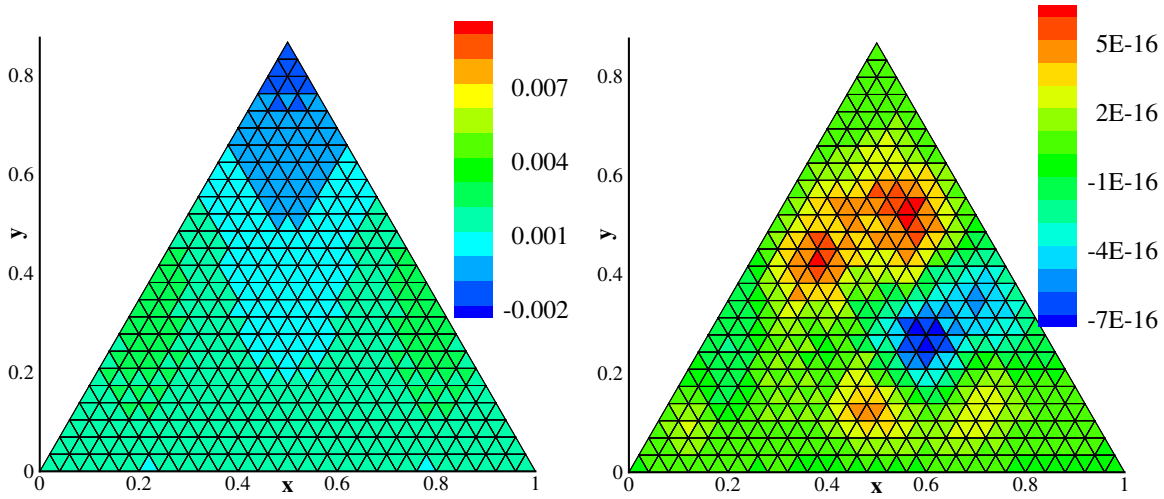


Figure 4.4: Poisson solution for the perfect mesh

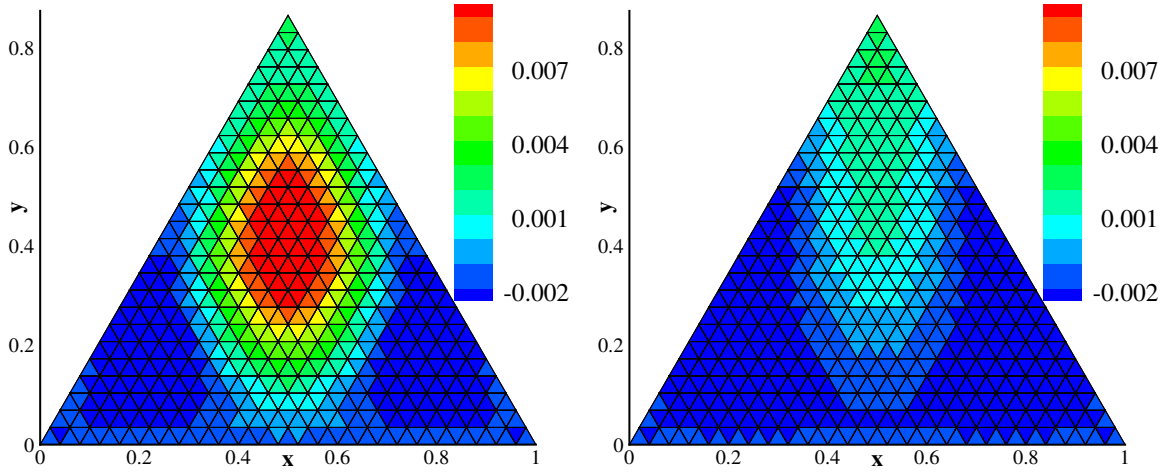
error is less compared to adding $L_3(U_2)$ as higher order terms are approximated.

The convergence of discretization error with mesh refinement is shown in Figure 4.6. Dashed lines are the discretization error of the original second, third and fourth order solution and the green line depicts the error for interior points of third order solution. As mentioned earlier, the third order discretization is second order accurate, however if just the interior cells are considered, the convergence is better than quadratic for the perfect mesh. Solid lines with square symbols are the defect corrected solutions based on second order solutions adding third and fourth order flux integrals as truncation error estimates. Despite the fact that the nominal expected convergence rate is third and fourth, both of them are quadratic. Although the magnitude of the error of the corrected solution by $L_4(U_2)$ is less than $L_3(U_2)$, supported by comparing Figures 4.5c and 4.5d, both of them are larger than the original non-corrected second order solution. On the other hand, correcting the third order solution by using fourth order flux integral, solid line with triangles, has a different behavior. The magnitude of the error is much smaller than the original third order solution and is approximately the same as the fourth order solution. Furthermore, it converges quartically which is the expected order. Therefore, for a perfect mesh, defect correction based on the third order solution is as good as expected; however, the performance of defect correction based on the second order solution is not good.



(a) Original problem

(b) Corrected by exact TE



(c) Corrected by $L_3\,(U_2)$

(d) Corrected by $L_4\,(U_2)$

Figure 4.5: Discretization error of the original and corrected solutions for perfect triangular mesh

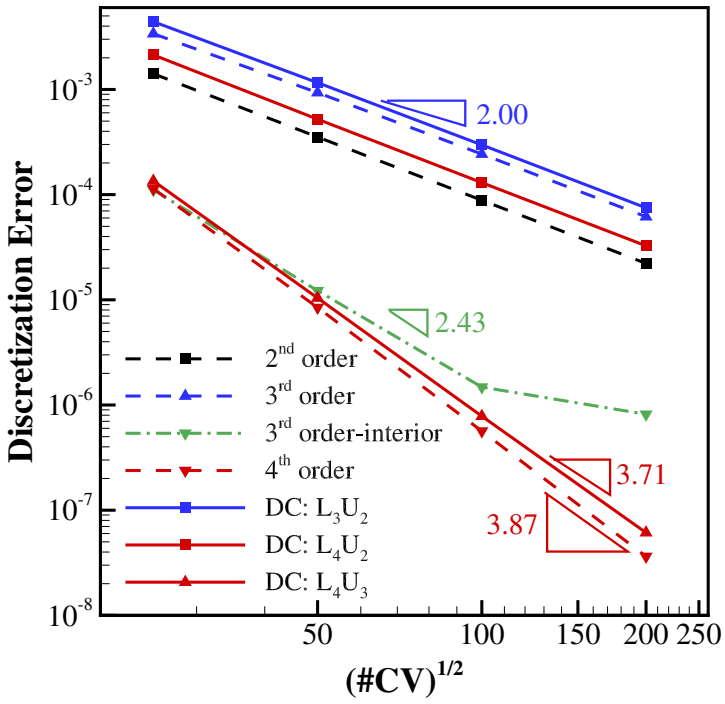


Figure 4.6: Defect correction based on p-TE method for perfect mesh

4.4 Defect Correction for an Unstructured Mesh

Having investigated the performance of defect correction on a perfect mesh, we study defect correction using the p-TE method on a general unstructured mesh for the same two model problems.

Figure 4.7 shows defect correction results for the Poisson problem; similar to the perfect mesh, defect correction based on the second order solution is not helpful and the magnitude of the error for the corrected solution is even more than the original second order problem. In contrast to the perfect mesh, defect correction based on the third order solution is not able to provide the nominal convergence rate. Although the magnitude of the error of the corrected solution is less than the third order solution, it is much larger than the error of the fourth order solution. Therefore, the performance of defect correction for a general unstructured mesh for diffusive fluxes are not as good as expected in terms of the convergence rate and magnitude of the error for the second order solution and in terms of convergence rate for the third order solution.

The same experiment is done for the advection problem and the results are shown

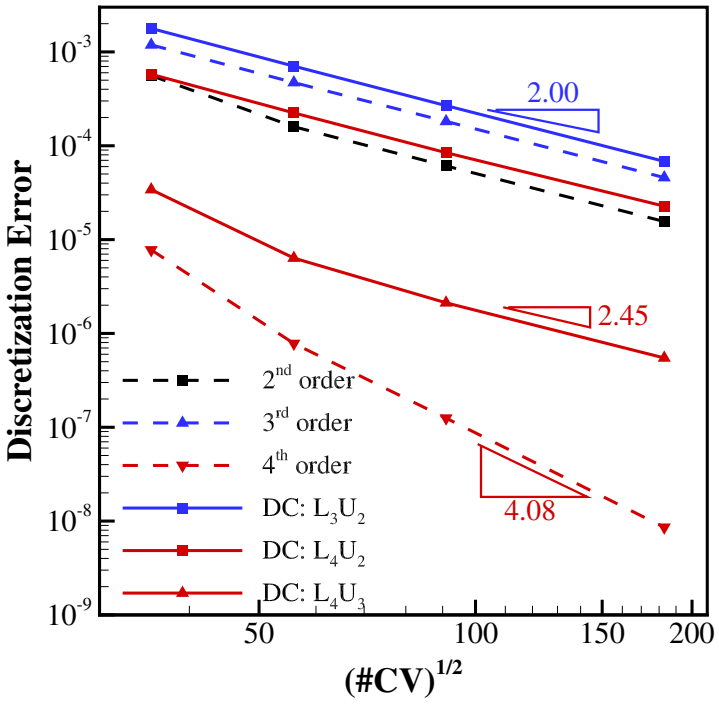


Figure 4.7: Defect correction based on p-TE method for Poisson on general unstructured mesh

in Figure 4.8. For convective fluxes, the error of the defect corrected solution based on the second order solution is less than the original second order one. The difference is more noticeable when the fourth order flux integral is added to the source term; however both converge quadratically and hence, the nominal order is not achieved. The same behavior exists for the corrected solution based on the third order solution. It converges cubically rather than quartically. Therefore, the performance of defect correction for convective fluxes is good in terms of error magnitude but not in terms of the convergence rate.

In conclusion, the expected convergence rate is not achievable by correcting the solutions based on the p-TE method either for diffusive fluxes or for convective fluxes. It should be noted that the same convergence results are obtained even on finer meshes as the solutions on the meshes we have shown here are in asymptotic range.

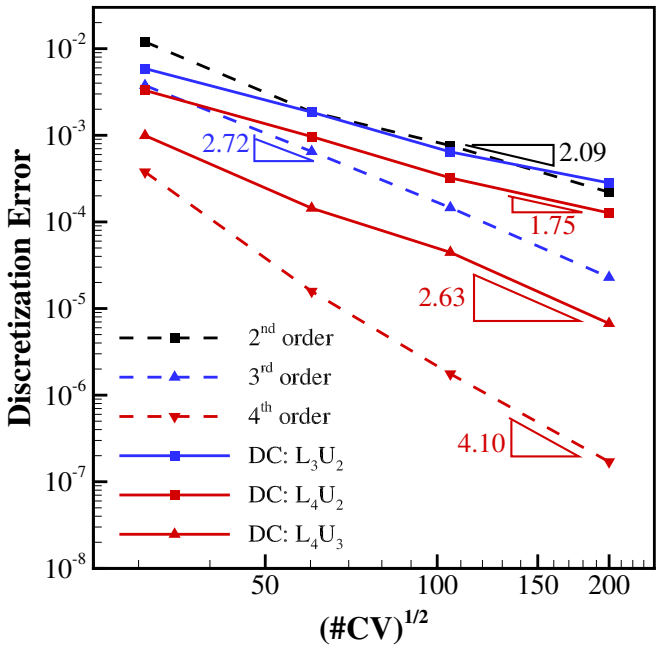


Figure 4.8: Defect correction based on p-TE method for advection on general unstructured mesh

4.5 Defect Correction Based on Continuous p-Truncation Error

Considering the fact that defect correction based on original p-TE method is not helpful for diffusive and convective fluxes, we try to use the continuous p-estimate of the truncation error to perform defect correction. Both CGM and C^1 interpolation are used for interpolating the solution and the results are shown for diffusive and convective fluxes.

4.5.1 Poisson

Our results based on the discrete p-TE method shows that the performance of defect correction is not as good as expected. Figure 4.9 depicts the results for the Poisson problem using continuous p-truncation error by both schemes discussed in the previous chapter where the dash-dotted lines show the original non-corrected solution, solid lines represent defect corrected solution based on discrete p-truncation error and the dashed lines are the corrected solution by the continuous p-truncation error, so $L_p^s U_q$ representing the continuous p-truncation error calculated by p order flux integration

based on q order converged solution.

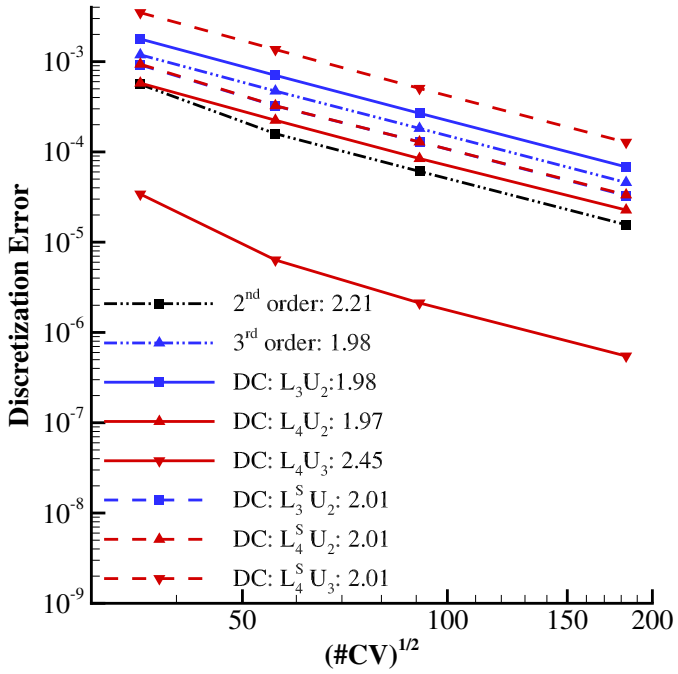
Using CGM for interpolating the second order solution corrects the solution such that the discretization error is less than the corrected solution by the discrete p-TE method; however, the corresponding discretization error is larger than the second-order non-corrected solution. Furthermore, both discretization errors of the corrected solutions by third and fourth order flux integrals based on the interpolated second order solution converge quadratically with mesh refinement which is not the expected order. Defect correction based on the third order interpolated solution by CGM is not helpful in terms of the magnitude of the error or the convergence rate, compared to the corrected solution based on the discrete p-truncation error.

Using C^1 interpolation of the solution based on second order solution, on the other hand, provides a corrected solution with smaller discretization error compared to the discrete solution. Moreover, the convergence rate is faster compared to the corrected solution based on discrete p-truncation error. Correction based on the third order solution is not helpful even with the continuous p-truncation error using C^1 interpolation of the solution.

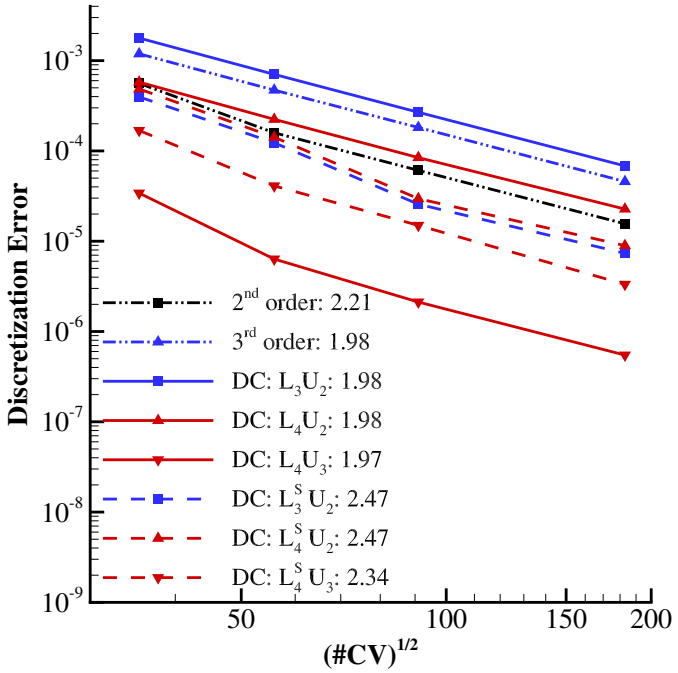
In conclusion, for the Poisson problem, defect correction based on continuous p-truncation error based on interpolated solution by CGM is not helpful neither for second order nor for third order solution. C^1 interpolation of the solution is able to provide corrected solutions with smaller discretization error when the second order solution is used and produces a convergence rate that is faster than quadratic.

4.5.2 Advection

The same experiment has been done for the advection problem and the results are shown in Figure 4.10. Interpolating by CGM for either the second order or the third order solution is not helpful in terms of convergence rate. Even though the convergence rate of the corrected solution by using the fourth order flux integral based on the interpolated second order solution is faster than quadratic, it is still significantly different from the nominal order. However, by using C^1 interpolation to obtain the continuous p-truncation error, the discretization error of the corrected solution is less than the discretization error of the corrected solution by discrete p-truncation error, even though the convergence rate is not as good as expected. The same behavior in terms of convergence rate exists for the third order solution.

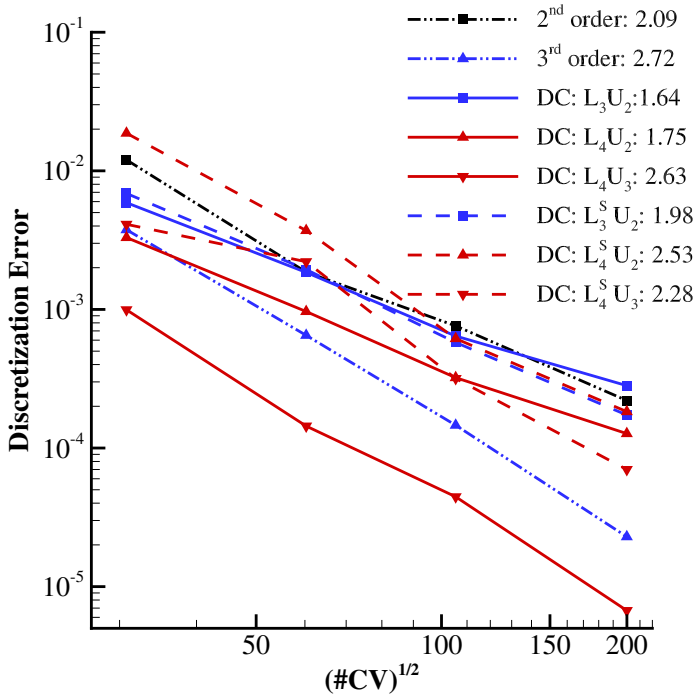


(a) Solution interpolation by CGM

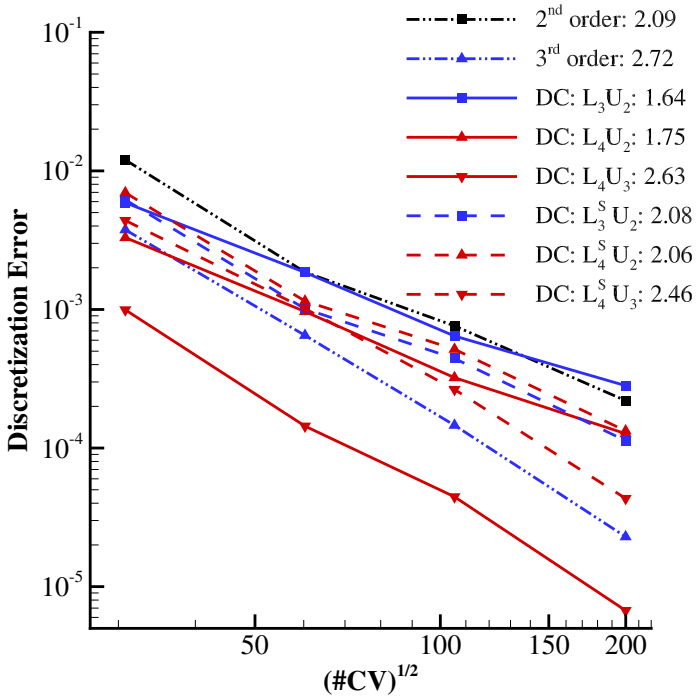


(b) C^1 interpolation of the solution

Figure 4.9: Defect correction results using continuous p-truncation error for Poisson



(a) Solution interpolation by CGM



(b) C^1 interpolation of the solution

Figure 4.10: Defect correction results using continuous p-truncation error for advection

4.5.3 Discussion

In summary, the performance of defect correction based on discrete p-truncation error is not helpful in terms of the convergence rate and the magnitude of the discretization error for diffusive and convective fluxes. Using CGM for interpolating the solution is not able to improve the performance of defect correction significantly either. For both diffusive and convective fluxes, using C^1 interpolation provides a corrected solution with smaller discretization error, but the converge rate is still not as good as expected. In particular, the performance is significantly better for correction based on the second-order solution than on the third-order solution.

As a consequence of poor performance of CGM for defect correction, we will continue with C^1 interpolation of the solution. CGM will not be used for interpolating in output error estimation and mesh adaptation discussed in the following chapters.

Although C^1 interpolation of the solution is not able to improve the performance of defect correction significantly, particularly in terms of convergence rate, this does not necessarily mean that it is not able to improve an output functional or effectively adapt the mesh. In the next two chapters, we will show the performance of output error estimation and mesh adaptation using the continuous p-truncation error by C^1 interpolation of the solution.

Chapter 5

Output Error Estimation and Correction

The adjoint theory was first presented in the context of linear algebra by using the algebraic equations obtained from the discretization of the original problem. This is the basis for the discrete adjoint approach. The continuous adjoint approach, on the other hand, is formulated based on the adjoint PDE which is discretized and solved independently [26]. In this chapter, we will start by the derivation of both discrete and continuous adjoint solutions. The continuous adjoint formulation is derived for different governing equations; scalar equations containing both convective and diffusive terms and system of equations for both Euler and Navier-Stokes.

The adjoint solution, either the discrete adjoint or continuous adjoint, can be multiplied by the truncation error to obtain the correction term used to correct the output functional of interest. Popular output functionals in aerodynamics are lift and drag. Both discrete and continuous p-truncation error may be weighted by the adjoint solution to obtain the correction term. If the higher-order flux integral is calculated based on the lower-order solution as an estimation of the truncation error and this estimate of the truncation error is weighted by the adjoint solution, the corrected output functional converges to the exact value as fast as using the higher-order solution directly.

The effect of using the discrete and continuous adjoint solution multiplied by both discrete and continuous p-truncation error in correcting the output functional is discussed in this chapter. The test cases include scalar equations, the advection and Poisson problem with exact solutions, and system of equations, Euler and Navier-Stokes using both manufactured solution and real-flows.

5.1 Discrete Adjoint

Consider a partial differential equation $R\left(\overline{U}\right) = 0$. This can be discretized — in our case using a finite volume discretization — and written as an algebraic equation:

$$R_h\left(\overline{U}_h\right) = 0 \tag{5.1}$$

Given a scalar output, $J_h\left(\overline{U}_h\right)$ such as lift or drag, the associated adjoint vector, Z_h, is the sensitivity of J_h to an infinitesimal residual perturbation, added to the nonlinear system:

$$\delta J_h \equiv J_h\left(\overline{U}_h + \delta\overline{U}_h\right) - J_h\left(\overline{U}_h\right) \equiv Z_h^T \delta R_h \tag{5.2}$$

where $\delta\overline{U}_h$ is the infinitesimal solution perturbation satisfying

$$\frac{\partial R_h}{\partial \overline{U}_h}\delta\overline{U}_h = \delta R_h \tag{5.3}$$

which is obtained by linearizing Eq. 5.1. The linearization assumes the discrete equations are differentiable. Furthermore, assuming that the output is also differentiable

$$\delta J_h = \frac{\partial J_h}{\partial \overline{U}_h}\delta\overline{U}_h = Z_h^T \delta R_h = -Z_h^T \frac{\partial R_h}{\partial \overline{U}_h}\delta\overline{U}_h \tag{5.4}$$

For this equation to hold for all perturbations requires that

$$\frac{\partial J_h}{\partial \overline{U}_h} = -Z_h^T \frac{\partial R_h}{\partial \overline{U}_h}$$

from which Z_h must satisfy the discrete adjoint equation:

$$\left(\frac{\partial R_h}{\partial \overline{U}_h}\right)^T Z_h + \left(\frac{\partial J_h}{\partial \overline{U}_h}\right)^T = 0 \tag{5.5}$$

As discussed in Chapter 1, the implicit Jacobian matrix can be computed in our solver and hence the transpose of the Jacobian matrix, $\left(\frac{\partial R_h}{\partial \overline{U}_h}\right)^T$, can be calculated easily. The discrete adjoint solution is used for mesh adaptation and optimization [59, 60].

If we assume that both the equations and the functional J have been linearized, the discrete approach can be described as a mapping from the original problem $J_h = (U_h, g)$ given that $AU_h = f$ into an equivalent adjoint problem $J_h = (Z_h, f)$ given that $A^T Z_h = g$. The inner product is a vector dot product $(V, U) = V^T U$ and the equivalence of the

two problems is easily proved to be

$$(Z_h, f) = (Z_h, AU_h) \equiv \left(A^T Z_h, U_h\right) = (g, U_h) \tag{5.6}$$

Note that the inhomogeneous term f in the discrete equations in the primal problem, enters the functional in the adjoint problem, and correspondingly the inhomogeneous term g in the adjoint problem comes from the functional of the primal problem.

5.2 Continuous Adjoint

The continuous primal problem can be posed as

$$\begin{array}{lrcll} \textit{Determine} & J & = & (U,g)_D + (CU,h)_{\partial D} & \\ \textit{given that} & LU & = & f & \textit{in } D \\ & BU & = & e & \textit{in } \partial D \end{array} \tag{5.7}$$

where $(U,V)_D = \int_D U^T V dv$ is an integral over the domain and $(U,V)_{\partial D} = \int_{\partial D} U^T V dA$ is an integral over the boundary of the domain. The objective in the analytic approach is to convert the primal problem using integration by parts into an equivalent adjoint problem

$$\begin{array}{lrcll} \textit{Determine} & J & = & (Z,f)_D + (C^* Z, e)_{\partial D} & \\ \textit{given that} & L^* Z & = & g & \textit{in } D \\ & B^* Z & = & h & \textit{on } \partial D \end{array} \tag{5.8}$$

where L^* is the linear PDE which is adjoint to L. B and B^* are boundary condition operators for the primal and adjoint problems, respectively, and C and C^* are (possibly differential) weight boundary operators used in the functionals; these four operators may have different dimensions on different parts of the boundary.

The two forms of the problem are equivalent provided that

$$J = (Z, LU)_D + (C^* Z, BU)_{\partial D} = (L^* Z, U)_D + (B^* Z, CU)_{\partial D} \tag{5.9}$$

which is the adjoint consistency condition. The left hand side is the output functional based on the adjoint solution and the right hand side is the output functional based on the primal problem. The continuous adjoint has been used for optimizing wing shape [40] and correcting the output functional [5, 82].

Comparing the discrete adjoint to the continuous adjoint reveals that although the discrete adjoint solves a system of equations based on the transpose of the Jacobian matrix and the derivative of the functional with respect to the solution, the continuous adjoint is an independent PDE which needs to be discretized and solved by itself. This implies that boundary conditions should be set for the continuous adjoint problem and those boundary conditions are dependent on the functional. In this section, the adjoint PDE, L^*, and the boundary conditions for different problems based on corresponding functionals are discussed. We start by examining convective and diffusive fluxes separately using the advection and Poisson problems, respectively. Hence, two model problems are defined by the method of manufactured solutions [76]. For systems of equations, both inviscid and viscous flows are considered.

5.2.1 Advection Equation

Consider the 2D steady linear advection equation with zero source term:

$$\begin{aligned} \nabla \cdot (\mathbf{b}U) &= 0 \; in \, D \\ U &= u_{in} \; on \, \partial D_i \end{aligned} \tag{5.10}$$

where $\mathbf{b}$ is the velocity vector and u_{in} is the solution at the inlet ∂D_i. The left hand side of Eq. 5.10 is multiplied by Z, integrated over the domain D and integrated by parts to obtain the continuous adjoint equation:

$$\left(\nabla \cdot (\mathbf{b}U), Z\right)_D + \left(U, -\mathbf{b} \cdot \hat{n} Z\right)_{\partial D_i} = \left(U, -\mathbf{b} \cdot \nabla Z\right)_D + \left(U, \mathbf{b} \cdot \hat{n} Z\right)_{\partial D_o} \tag{5.11}$$

where $\hat{n}$ is the outward normal vector. Comparing to Eq. 5.9, we see that $LU = \nabla \cdot (\mathbf{b}U)$ in D and

$$\begin{aligned} BU &= U, \quad CU = 0 \quad on \, \partial D_i \\ BU &= 0, \quad CU = U \quad on \, \partial D_o \end{aligned}$$

for the primal problem and $L^*Z = -\mathbf{b} \cdot \nabla Z$ in D and

$$\begin{aligned} B^*Z &= 0, \quad C^*Z = -\mathbf{b} \cdot \hat{n} Z \quad on \, \partial D_i \\ B^*Z &= \mathbf{b} \cdot \hat{n} z, \quad C^*Z = 0 \quad on \, \partial D_o \end{aligned}$$

for the continuous adjoint problem. The output functional is defined as

$$\begin{aligned} J &= \int_D (\nabla \cdot (\mathbf{b}U))\, Z dA + \int_{\partial D_i} u_{in}\, (-\mathbf{b} \cdot \hat{n} Z)\, ds \\ &= \int_D (-\mathbf{b} \cdot \nabla Z)\, U dA + \int_{\partial D_o} (\mathbf{b} \cdot \hat{n} Z)\, u ds \end{aligned} \tag{5.12}$$

For a constant velocity vector, the adjoint equation to the advection problem is an advection equation in the reverse direction.

5.2.2 Poisson Equation

Consider the 2D Poisson equation:

$$\begin{aligned} \triangle U &= f \ in\, D \\ U &= b_D \ on\, \partial D_D \\ \hat{n} \cdot \nabla U &= b_N \ on\, \partial D_N \end{aligned} \tag{5.13}$$

where b_D is the Dirichlet boundary condition on ∂D_D and b_N is the Neumann boundary condition on ∂D_N, with $\partial D_D \cup \partial D_N = \partial D$ and $\partial D_D \cap \partial D_N = 0$. Similar to the advection problem, the left hand side of Eq. 5.13 is multiplied by Z, integrated over the domain D and integrated by parts to obtain the continuous adjoint equation:

$$\begin{aligned} (\triangle U, Z)_D &= -(\nabla U, \nabla Z)_D + (\hat{n} \cdot \nabla U, Z)_{\partial D} \\ &= (U, \triangle Z)_D - (U, \hat{n} \cdot \nabla Z)_{\partial D} + (\hat{n} \cdot \nabla U, Z)_{\partial D} \end{aligned} \tag{5.14}$$

splitting the boundary terms according to $\partial D_D \cup \partial D_N = \partial D$, and shuffling terms:

$$\begin{aligned} &(\triangle U, Z)_D + (U, \hat{n} \cdot \nabla Z)_{\partial D_D} + (\hat{n} \cdot \nabla U, -Z)_{\partial D_N} = \\ &(U, \triangle Z)_D + (\hat{n} \cdot \nabla U, Z)_{\partial D_D} + (U, -\hat{n} \cdot \nabla Z)_{\partial D_N} \end{aligned}$$

Comparing to Eq. 5.9, we end up with $LU = \triangle U$ in D and

$$\begin{aligned} BU &= U, & CU &= \hat{n} \cdot \nabla U & on\, \partial D_D \\ BU &= \hat{n} \cdot \nabla U, & CU &= U & on\, \partial D_N \end{aligned}$$

for the primal problem and $L^* Z = \triangle Z$ in D and

$$\begin{aligned} B^*Z &= Z, & C^*Z &= \hat{n}\cdot\nabla Z \quad on\, \partial D_D \\ B^*Z &= -\hat{n}\cdot\nabla Z, & C^*Z &= -Z \quad on\, \partial D_N \end{aligned}$$

for the continuous adjoint problem. Comparing L and L^* shows that the primal and adjoint operators are the same and as a result, Poisson is self adjoint implying that the Poisson adjoint problem is a Poisson problem itself for which the source term and boundary conditions are defined based on the functional of interest. The output functional is defined as:

$$\begin{aligned} J &= \int_D \triangle U Z dA + \int_{\partial D_D} U\left(\hat{n}\cdot\nabla Z\right) ds + \int_{\partial D_N} \left(\hat{n}\cdot\nabla U\right)(-Z)\, ds \\ &= \int_D \triangle Z U dA + \int_{\partial D_D} \left(\hat{n}\cdot\nabla U\right) Z ds + \int_{\partial D_N} \left(-\hat{n}\cdot\nabla Z\right) U ds \end{aligned} \tag{5.15}$$

5.2.3 Euler Equations

Consider the linearized steady-state Euler equations:

$$LU = \frac{\partial}{\partial x}\left(A_x U\right) + \frac{\partial}{\partial y}\left(A_y U\right) = f \tag{5.16}$$

where $U = \begin{pmatrix} \rho & \rho u & \rho v & \rho e \end{pmatrix}^T$ is the conserved solution vector and $A_x = \frac{\partial F_x^c}{\partial U}$ and $A_y = \frac{\partial F_y^c}{\partial U}$ are the flux Jacobian in which F_x and F_y are the convective flux functions:

$$F_x^c = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uh \end{pmatrix}, \quad F_y^c = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vh \end{pmatrix}. \tag{5.17}$$

where $h = e + \frac{p}{\rho} = \frac{P\gamma}{\rho(\gamma-1)} + \frac{1}{2}\left(u^2+v^2\right)$ is the enthalpy and $\gamma = \frac{C_p}{C_v}$ is the ratio of specific heats. Consequently, the flux Jacobians are:

$$A_x = \frac{\partial F_x}{\partial U} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ \frac{\gamma-1}{2}\left(u^2+v^2\right) - u^2 & (3-\gamma)\, u & (1-\gamma)\, v & \gamma - 1 \\ -uv & v & u & 0 \\ u\left(\frac{\gamma-1}{2}\left(u^2+v^2\right) - h\right) & h - (\gamma-1)\, u^2 & (1-\gamma)\, uv & \gamma u \end{pmatrix}$$

$$A_y = \frac{\partial F_y}{\partial U} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ -uv & v & u & 0 \\ \frac{\gamma-1}{2}\left(u^2+v^2\right)-v^2 & (1-\gamma)\,u & (3-\gamma)\,v & \gamma-1 \\ v\left(\frac{\gamma-1}{2}\left(u^2+v^2\right)-h\right) & (1-\gamma)\,uv & h-(\gamma-1)\,v^2 & \gamma v \end{pmatrix}$$

Note that we are using Roe's scheme [77] for evaluating the flux function. Using this scheme, the flux function at the face between cell i and j is evaluated as:

$$F\left(U_i, U_j\right) = \frac{1}{2}\left[\left(F\left(U_i\right)+F\left(U_j\right)\right) - \left|\widetilde{A}\right|\left(U_j - U_i\right)\right]. \tag{5.18}$$

$\widetilde{A}$ is the Jacobian matrix evaluated based on the Roe's average properties defined as:

$$\begin{aligned} \widetilde{\rho} &= \sqrt{\rho_i \rho_j} \\ \widetilde{u} &= \frac{\sqrt{\rho_i}u_i + \sqrt{\rho_j}u_j}{\sqrt{\rho_i}+\sqrt{\rho_j}} \\ \widetilde{v} &= \frac{\sqrt{\rho_i}v_i + \sqrt{\rho_j}v_j}{\sqrt{\rho_i}+\sqrt{\rho_j}} \\ \widetilde{h} &= \frac{\sqrt{\rho_i}h_i + \sqrt{\rho_j}h_j}{\sqrt{\rho_i}+\sqrt{\rho_j}} \end{aligned} \tag{5.19}$$

and $\left|\widetilde{A}\right|$ can be written in diagonalized form as:

$$\left|\widetilde{A}\right| = \widetilde{X}^{-1}\left|\widetilde{\Lambda}\right|\widetilde{X}. \tag{5.20}$$

$\widetilde{X}$ is a matrix whose columns are the right eigenvectors and the components of $\widetilde{\Lambda}$ are the eigenvalues of the Jacobian matrix:

$$\left|\widetilde{\Lambda}\right| = \begin{pmatrix} |\widetilde{u}_n| & 0 & 0 & 0 \\ 0 & |\widetilde{u}_n| & 0 & 0 \\ 0 & 0 & \left|\widetilde{u}_n + \widetilde{C}\right| & 0 \\ 0 & 0 & 0 & \left|\widetilde{u}_n - \widetilde{C}\right| \end{pmatrix} \tag{5.21}$$

where $\widetilde{u}_n = n_x\widetilde{u} + n_y\widetilde{v}$ is the normal velocity and $\widetilde{C} = \sqrt{(\gamma-1)\left(\widetilde{h} - \frac{1}{2}\left(\widetilde{u}^2 + \widetilde{v}^2\right)\right)}$ is the sound velocity.

To obtain the continuous adjoint equation, Eq. 5.16 is multiplied by Z, integrated over the domain D and integrated by parts:

$$\begin{aligned}\left(\frac{\partial}{\partial x}\left(A_x U\right) + \frac{\partial}{\partial y}\left(A_y U\right), Z\right)_D &= \left(U, -A_x^T\frac{\partial Z}{\partial x} - A_y^T\frac{\partial Z}{\partial y}\right)_D \\ &+ \left(\left(n_x A_x + n_y A_y\right)U, Z\right)_{\partial D}.\end{aligned} \tag{5.22}$$

Comparing to Eq. 5.9, we find that

$$L^* Z = -A_x^T\frac{\partial Z}{\partial x} - A_y^T\frac{\partial Z}{\partial y} \tag{5.23}$$

for the continuous adjoint problem. Note that the primal PDE, Eq. 5.16, is written in the conservative form; however, the adjoint PDE, Eq. 5.23 is not in conserved form and we need to transform that to the conserved form; this transformation is described in detail in Appendix B. The continuous adjoint fluxes are calculated by the Lax-Friedrich method based on the maximum eigenvalue of the Euler Jacobian:

$$F\left(U_i, U_j\right) = \frac{1}{2}\left[\left(F\left(U_i\right) + F\left(U_j\right)\right) - \lambda_{max}\left(U_j - U_i\right)\right] \tag{5.24}$$

To be able to solve the continuous adjoint equation, we need to define the boundary operators for the solid wall where the output functional is defined, as well as supersonic and subsonic inflow and outflow.

Solid Wall

The output functional corresponding to drag or lift is:

$$J = \int_{\partial D_{wall}} p\hat{n}\cdot\psi ds \tag{5.25}$$

where $\psi = (\cos\alpha, \sin\alpha)^T$ for drag and $\psi = (-\sin\alpha, \cos\alpha)^T$ for lift, and α is the angle of attack. Using the primal boundary operator in the right hand side of Eq. 5.22 for the slip wall boundary condition, $\vec{u}\cdot\hat{n} = un_x + vn_y = 0$, and multiplying by the adjoint

solution, we obtain:

$$\begin{aligned}((n_x A_x + n_y A_y)\, U, Z)_{\partial D} &= \left(\begin{bmatrix} 0 & pn_x & pn_y & 0 \end{bmatrix}^T, \begin{bmatrix} Z_1 & Z_2 & Z_3 & Z_4 \end{bmatrix}^T\right) \\ &= pn_x Z_2 + pn_y Z_3\end{aligned}$$

Comparing to the functional of interest, Eq. 5.25, we obtain the continuous adjoint boundary condition as

$$n_x Z_2 + n_y Z_3 = \hat{n} \cdot \psi \tag{5.26}$$

Supersonic inflow/outflow

The solution components are set for supersonic inflow for the primal problem and for supersonic outflow, they are obtained by reconstruction. Eq. 5.23 shows that the characteristic behavior of the adjoint problem is similar to that of the primal Euler equation, but with the sign of each characteristic velocity reversed so that the characteristic information travels in the opposite direction, similar to the advection problem as described. All characteristics are coming into the domain for supersonic inflow, and so for the adjoint state, the characteristics are coming out and hence no boundary condition is set at the supersonic inflow for the adjoint problem and it is obtained by reconstruction directly. For the supersonic outflow, on the other hand, since for most applied problems, the output functional is defined as the lift or drag on the solid wall and the outflow does not have any contribution in the functional, the boundary condition for the adjoint problem is zero at the supersonic outflow [5]. A supersonic outflow has no influence on any quantity.

Subsonic inflow/outflow

At a subsonic inflow boundary, there are three characteristics entering the domain for the primal problem and therefore, there is only one adjoint characteristic that leaves the domain. Accordingly, one boundary condition should be specified for adjoint subsonic inflow. It can be written in a simplified form as:

$$C_1 Z_1 + C_2 Z_2 + C_3 Z_3 + C_4 Z_4 = 0 \tag{5.27}$$

where the coefficients are given by [34]:

$$
\begin{aligned}
C_1 &= \frac{n_x u + n_y v}{2}\left(2 - \gamma + \gamma^2 - 2\frac{\gamma e\,(\gamma-1)}{u^2+v^2}\right) \\
C_2 &= \frac{\left(2-\gamma+\gamma^2\right) n_x u^2 + 2\left(1-\gamma+\gamma^2\right) n_y uv}{2} - \frac{(\gamma-1)\,\gamma n_x v^2}{2} \\
&\quad - \frac{\gamma e\,(\gamma-1)\left(2n_y uv + n_x\left(u^2-v^2\right)\right)}{\left(u^2+v^2\right)} \\
C_3 &= \frac{-\gamma\,(\gamma-1)\,n_y u^4 + 2n_y u^2 v^2}{2\left(u^2+v^2\right)} + \frac{\left(1-\gamma+\gamma^2\right) n_x\left(u^3 v + uv^3\right)}{\left(u^2+v^2\right)} \\
&\quad + \frac{\left(2-\gamma+\gamma^2\right) n_y v^4 + 2\gamma e\,(\gamma-1)\left(n_y\left(u^2-v^2\right) - 2n_x uv\right)}{2\left(u^2+v^2\right)} \\
C_4 &= \frac{\left(n_x u + n_y v\right)\left((\gamma-1)\left(u^2+v^2\right) - 2\gamma e\right)\left(\left(2-\gamma+\gamma^2\right)\left(u^2+v^2\right) - 2\gamma e\,(\gamma-1)\right)}{-4\left(u^2+v^2\right)}
\end{aligned}
$$

In the case of a subsonic outflow boundary, the adjoint problem has three incoming characteristics entering the domain implying that three boundary conditions are needed. They can be written as:

$$
\begin{aligned}
Z_1 &= \frac{1}{2}\left(2e\gamma - (\gamma-1)\left(u^2+v^2\right)\right) Z_4 \\
Z_2 &= \left(\frac{-2e\gamma n_x + (\gamma-2)\,n_x u^2 - 2n_y uv + \gamma n_x v^2}{2\left(n_x u + n_y v\right)}\right) Z_4 \\
Z_3 &= \left(\frac{-2e\gamma n_y + (\gamma-2)\,n_y v^2 - 2n_x uv + \gamma n_y u^2}{2\left(n_x u + n_y v\right)}\right) Z_4
\end{aligned}
$$

Table 5.1 summarizes the number of boundary conditions required for different boundaries according to the characteristics.

	Primal	Adjoint
Supersonic Inflow	4	0
Supersonic Outflow	0	4
Subsonic Inflow	3	1
Subsonic Outflow	1	3

Table 5.1: Number of required boundary conditions

5.2.4 Navier-Stokes Equations

Consider the linearized steady-state Navier-Stokes equations for constant viscosity:

$$\begin{aligned} LU &= \frac{\partial}{\partial x}\left(A_x U - D_{xx}\frac{\partial U}{\partial x} - D_{xy}\frac{\partial U}{\partial y}\right) \\ &+ \frac{\partial}{\partial y}\left(A_y U - D_{yx}\frac{\partial U}{\partial x} - D_{yy}\frac{\partial U}{\partial y}\right) = f \end{aligned} \tag{5.28}$$

where A_x and A_y are the derivatives of the convective flux function, Eq. 5.17, with respect to the solution as defined before and the other D_{ij} matrices are the derivatives of viscous flux with respect to the solution gradients

$$D_{xx} = \frac{\partial F_x^v}{\partial\left(\frac{\partial U}{\partial x}\right)} \quad , \quad D_{yx} = \frac{\partial F_y^v}{\partial\left(\frac{\partial U}{\partial x}\right)}$$

$$D_{xy} = \frac{\partial F_x^v}{\partial\left(\frac{\partial U}{\partial y}\right)} \quad , \quad D_{yy} = \frac{\partial F_y^v}{\partial\left(\frac{\partial U}{\partial y}\right)}$$

where F^v represents the viscous flux function:

$$F_x^v = \begin{pmatrix} 0 \\ \frac{Ma}{Re}\mu\left(\frac{4}{3}\frac{\partial u}{\partial x} - \frac{2}{3}\frac{\partial v}{\partial y}\right) \\ \frac{Ma}{Re}\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \\ \frac{Ma}{Re}\mu\left(u\left(\frac{4}{3}\frac{\partial u}{\partial x} - \frac{2}{3}\frac{\partial v}{\partial y}\right) + v\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right) + \frac{\mu\gamma Ma}{RePr(\gamma-1)}\left(\frac{\rho\frac{\partial P}{\partial x} - P\frac{\partial \rho}{\partial x}}{\rho^2}\right) \end{pmatrix}$$

$$F_y^v = \begin{pmatrix} 0 \\ \frac{Ma}{Re}\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \\ \frac{Ma}{Re}\mu\left(\frac{4}{3}\frac{\partial v}{\partial y} - \frac{2}{3}\frac{\partial u}{\partial x}\right) \\ \frac{Ma}{Re}\mu\left(u\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) + v\left(\frac{4}{3}\frac{\partial v}{\partial y} - \frac{2}{3}\frac{\partial u}{\partial x}\right)\right) + \frac{\mu\gamma Ma}{RePr(\gamma-1)}\left(\frac{\rho\frac{\partial P}{\partial y} - P\frac{\partial \rho}{\partial y}}{\rho^2}\right) \end{pmatrix}$$

where $Ma = \frac{\sqrt{u^2+v^2}}{C}$ is the Mach number, μ is the dynamic viscosity coefficient, Re is the Reynolds number and Pr is the Prandtl number. Consequently, the D_{ij} matrices,

the derivatives of viscous fluxes with respect to the solution gradients are:

$$D_{xx} = \frac{\mu}{\rho}\begin{pmatrix} 0 & 0 & 0 & 0 \\ -\frac{4}{3}u & \frac{4}{3} & 0 & 0 \\ -v & 0 & 1 & 0 \\ -\left(\frac{4}{3}u^2 + v^2 + \frac{\gamma}{Pr}\left(e - u^2 - v^2\right)\right) & \left(\frac{4}{3} - \frac{\gamma}{Pr}\right)u & \left(1 - \frac{\gamma}{Pr}\right)v & \frac{\gamma}{Pr} \end{pmatrix}$$

$$D_{xy} = \frac{\mu}{\rho}\begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{2}{3}v & 0 & -\frac{2}{3} & 0 \\ -u & 1 & 0 & 0 \\ -\frac{1}{3}uv & v & -\frac{2}{3}u & 0 \end{pmatrix}, \quad D_{yx} = \frac{\mu}{\rho}\begin{pmatrix} 0 & 0 & 0 & 0 \\ -v & 0 & 1 & 0 \\ \frac{2}{3}u & -\frac{2}{3} & 0 & 0 \\ -\frac{1}{3}uv & -\frac{2}{3}v & u & 0 \end{pmatrix}$$

$$D_{yy} = \frac{\mu}{\rho}\begin{pmatrix} 0 & 0 & 0 & 0 \\ -u & 1 & 0 & 0 \\ -\frac{4}{3}v & 0 & \frac{4}{3} & 0 \\ -\left(u^2 + \frac{4}{3}v^2 + \frac{\gamma}{Pr}\left(e - u^2 - v^2\right)\right) & \left(1 - \frac{\gamma}{Pr}\right)u & \left(\frac{4}{3} - \frac{\gamma}{Pr}\right)v & \frac{\gamma}{Pr} \end{pmatrix}$$

Eq. 5.28 is multiplied by Z, integrated over the domain D and integrated by parts to obtain the continuous adjoint equation:

$$\begin{aligned} \left(\frac{\partial}{\partial x}\left(A_x U - D_{xx}\frac{\partial U}{\partial x} - D_{xy}\frac{\partial U}{\partial y}\right) + \frac{\partial}{\partial y}\left(A_y U - D_{yx}\frac{\partial U}{\partial x} - D_{xy}\frac{\partial U}{\partial y}\right), Z\right)_D &= \\ \left(-A_x\frac{\partial Z}{\partial x} - \frac{\partial}{\partial x}\left(D_{xx}^T\frac{\partial Z}{\partial x} + D_{yx}^T\frac{\partial Z}{\partial y}\right) - A_y\frac{\partial Z}{\partial y} - \frac{\partial}{\partial y}\left(D_{xy}^T\frac{\partial Z}{\partial x} + D_{yy}^T\frac{\partial Z}{\partial y}\right), U\right)_D &+ \\ \left(n_x\left(A_x U - D_{xx}\frac{\partial U}{\partial x} - D_{xy}\frac{\partial U}{\partial y}\right) + n_y\left(A_y U - D_{yx}\frac{\partial U}{\partial x} - D_{yy}\frac{\partial U}{\partial y}\right), Z\right)_{\partial D} &+ \\ \left(\frac{\partial Z}{\partial x}, \left(n_x D_{xx} + n_y D_{yx}\right)U\right)_{\partial D} + \left(\frac{\partial Z}{\partial y}, \left(n_x D_{xy} + n_y D_{yy}\right)U\right)_{\partial D} & \end{aligned} \tag{5.29}$$

Comparing to Eq. 5.9, we end up with

$$\begin{aligned} L^* Z \;=\; & -A_x^T\frac{\partial Z}{\partial x} - A_y^T\frac{\partial Z}{\partial y} \\ & -\frac{\partial}{\partial x}\left(D_{xx}^T\frac{\partial Z}{\partial x} + D_{yx}^T\frac{\partial Z}{\partial y}\right) - \frac{\partial}{\partial y}\left(D_{xy}^T\frac{\partial Z}{\partial x} + D_{yy}^T\frac{\partial Z}{\partial y}\right) \end{aligned} \tag{5.30}$$

for the continuous adjoint problem where the first term on right-hand side is the same as the adjoint PDE for the Euler equation and the second term is already in conserved

form.

The most popular output functional of interest in viscous compressible flows are the total (i.e., pressure plus viscous) drag and lift. So, the output functional is defined as:

$$J = \int_{\partial D_{wall}} (p\hat{n} - \tau\hat{n}) \cdot \vec{\psi} ds = \int_{\partial D_{wall}} (pn_i - \tau_{ij} n_j)\, \psi_i ds \tag{5.31}$$

where ψ is the same as defined above and the viscous stress term is defined by

$$\tau = \begin{pmatrix} \frac{4}{3}\frac{\partial u}{\partial x} - \frac{2}{3}\frac{\partial v}{\partial y} & \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} & \frac{4}{3}\frac{\partial v}{\partial y} - \frac{2}{3}\frac{\partial u}{\partial x} \end{pmatrix} \tag{5.32}$$

Using the primal boundary operator in the right hand side of Eq. 5.29 for the adiabatic no-slip wall boundary condition

$$\begin{aligned} u &= 0 \\ v &= 0 \\ \hat{n} \cdot \nabla T &= 0 \end{aligned} \tag{5.33}$$

and multiplying by the adjoint solution, we obtain the continuous adjoint boundary condition for the adiabatic no-slip wall:

$$\begin{aligned} Z_2 &= \psi_1 \\ Z_3 &= \psi_2 \\ \hat{n} \cdot \nabla Z_4 &= 0 \end{aligned} \tag{5.34}$$

The convective part of the boundary condition for the inflows and outflows are the same as the continuous adjoint for the Euler problem.

5.3 Output Functional Correction

Pierce and Giles have developed a method to estimate error in the functional [73] and an overview of the method is discussed here. Considering Eq. 5.9 and for given continuous

approximate solutions U_h and Z_h of the discrete solution:

$$\begin{aligned} J &= (g,U)_D + (h,CU)_{\partial D} \\ &= (g,U_h+(U-U_h))_D + (h,C(U_h+(U-U_h)))_{\partial D} \\ &= (L^*Z_h,U_h)_D + (L^*Z_h,U-U_h)_D + (g-L^*Z_h,U-U_h)_D + (g-L^*Z_h,U_h)_D \\ &+ (B^*Z_h,CU_h)_{\partial D} + (B^*Z_h,C(U-U_h))_{\partial D} \\ &+ (h-B^*Z_h,C(U-U_h))_{\partial D} + (h-B^*Z_h,CU_h)_{\partial D} \end{aligned} \tag{5.35}$$

By using integration by parts, we can re-write the above equation as

$$\begin{aligned} J &= (L^*Z_h,U_h)_D + (B^*Z_h,CU_h)_{\partial D} + \\ &\quad (Z_h,L(U-U_h))_D + (C^*Z_h,B(U-U_h))_{\partial D} + \\ &\quad (Z-Z_h,L(U-U_h))_D + (C^*(Z-Z_h),B(U-U_h))_{\partial D} \end{aligned} \tag{5.36}$$

The two terms in the first line are the influence of bulk and boundary integrals in the discrete functional, the terms in the second line represent computable adjoint error estimates and the terms in the third line are the higher order remaining errors which are negligible compared to other terms. Accordingly, the estimated error in calculating the functional is:

$$\begin{aligned} \delta J &\approx (Z_h,L(U-U_h))_D + (C^*Z_h,B(U-U_h))_{\partial D} \\ &= (Z_h,f-LU_h)_D + (C^*Z_h,e-BU_h)_{\partial D} \\ &= (Z_h,\tau)_D + (C^*Z_h,e-BU_h)_{\partial D} \end{aligned} \tag{5.37}$$

where τ is the primal truncation error. Hence, an estimate of the truncation error is needed to find the correction term for the output functional. The estimate of the truncation error based on the interpolated solution is multiplied by the adjoint solution to find the correction term added to the output functional to have a more accurate functional which converges to the exact value with a higher-order rate. It should be noted that the correction term of Eq. 5.37 is the summation over all control volumes of the truncation error estimated by computing the higher-order flux integral based on the lower-order solution multiplied by the adjoint solution of the same order as the primal problem

$$\delta J = (Z_h,\tau)_D = \sum_{CV}\left(\oint \vec{F}_p^{p+1}\cdot\hat{n}ds\right)\bar{Z}_p$$

The higher-order flux integration can be computed to order $p+2$ as well.

We will compare the effectiveness of using the original discrete p-TE estimate and the continuous p-TE estimate for different governing equations with different physical behavior, for both convective and diffusive fluxes. Both scalar and system problems are considered. For each case, the domain of the problem is introduced and then corrections based on the discrete and continuous p-truncation error are compared for both discrete and continuous adjoint problems.

To verify our approach to improving the functional, we use problems with exact solutions so that we can calculate the exact functional to compare the magnitude of the error and also the convergence rate of the corrected functional with mesh refinement. Since this technique is mathematical in origin rather than physical, applying it to other problems, with or without exact solutions, follows exactly the same procedure. We are focusing on the problems with exact solutions only to have an exact reference value for comparison purposes.

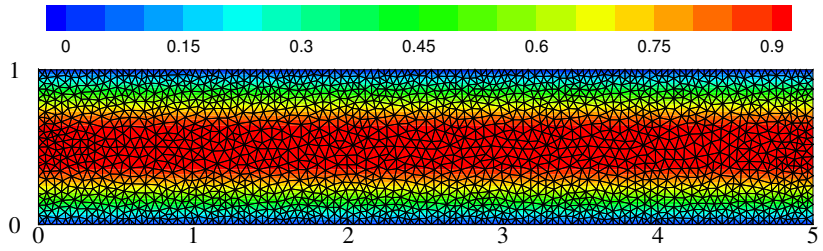
5.4 Advection Equation

The velocity vector in Eq. 5.10 is set as $\mathbf{b} = (1, 0)$ and the primal problem is the same as the advection problem described in Chapter 4, Figure 4.3. The problem domain considered is a (5×1) rectangular channel and the output functional is defined as

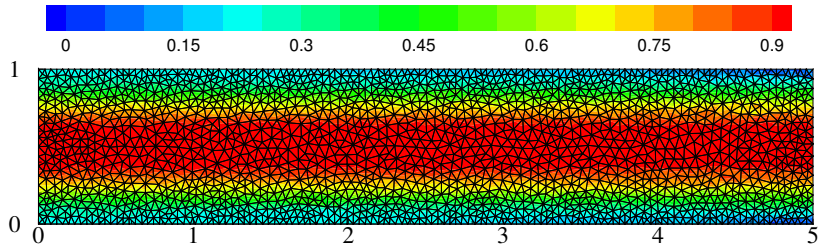
$$J = (U, g)_D + (CU, h)_{\partial D_0} = 0 + (U, h)_{\partial D_0} = \int_0^1 \sin(\pi y) \sin(\pi y)\, dy = 0.5$$

implying that the adjoint problem boundary condition should be set as $\sin(\pi y)$. The functional is defined intentionally such that the adjoint problem has exactly the same distribution as the primal problem. Both continuous and discrete adjoint solutions are depicted in Figure 5.1 and as their distributions should be the same as the the exact primal problem, so comparing continuous and discrete adjoint is easier. Although both solutions are qualitatively similar to the desired solution, the continuous adjoint is smoother and more accurate, as expected, as a consequence of solving the continuous adjoint PDE instead of solving the transposed Jacobian system arising from the discrete solution. This increased accuracy is reflected in the correction process as well.

Before showing the numerical results, the nomenclature we are using should be described; J is the functional, CJ is the corrected functional and C's superscript rep-



(a) Continuous adjoint

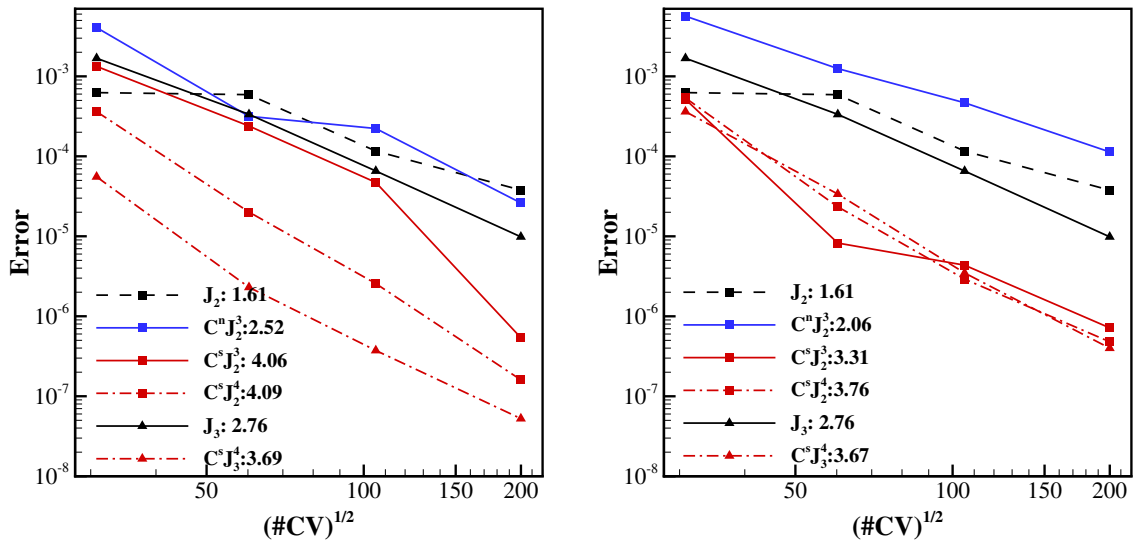


(b) Discrete adjoint

Figure 5.1: Adjoint solutions for advection problem

resents whether the correction is based on truncation error using the discrete solution, C^n, or is based on the truncation error using the interpolated solution, C^s. J's subscript shows the primal solution order of accuracy and the superscript shows the order of reconstruction and flux integration used as an estimation of the truncation error.

The error in calculating the functional is depicted in Figure 5.2 using both continuous and discrete adjoint solutions. Comparing the functional based on the second-order solution with the corrected functional based on the discrete solution shows poor performance of the correction in terms of error magnitude and also asymptotic convergence rate for both continuous and discrete adjoint solutions. On the other hand, correcting the functional by using the truncation error based on the interpolated solution, $C^s J_2^3$, has smaller error compared to the second-order functional and the rate of convergence is 4.06 and 3.31 for continuous and discrete adjoint, respectively. In this case, we have super-convergence with the continuous adjoint and the convergence is quartic instead of cubic. Using fourth order flux integration for estimating the truncation error based on the interpolated second-order solution, $C^s J_2^4$, further reduces the error. Comparing the error magnitudes of these corrected solutions based on second-order solution with the error of the third-order solution reveals that this correction scheme based on the interpolated solution is even more accurate than the higher-order functional. Instead



(a) Continuous adjoint

(b) Discrete adjoint

Figure 5.2: Convergence history for the advection problem

of using the second-order functional, the third-order functional may be corrected based on fourth-order flux integration, $C^s J_3^4$. Similarly, the convergence is quartic for the corrected solution although the original functional is cubic.

```
Since the same range is used for errors for both continuous and discrete adjoints
```

5.5 Poisson Equation

The primal problem is defined on a (1×1) square domain as

$$\begin{aligned} \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} &= -\frac{\pi^3}{4} \sin(\pi x) \sin(\pi y), \; U = \sin(\pi x) \; on \; y = 1 \\ \Rightarrow U_{exact} &= \frac{\pi}{8} \sin(\pi x) \sin(\pi y) + \frac{1}{\sinh(\pi)} \sin(\pi x) \sinh(\pi y) \end{aligned} \tag{5.38}$$

and the exact solution is shown in Figure 5.3.

Setting the adjoint source term as $g = \frac{\pi^5}{2} x(1-x)\, y(1-y)$, the adjoint boundary condition is obtained as $h = x(1-x)$ on $y = 0$ and zero elsewhere by Eq. 5.9. The

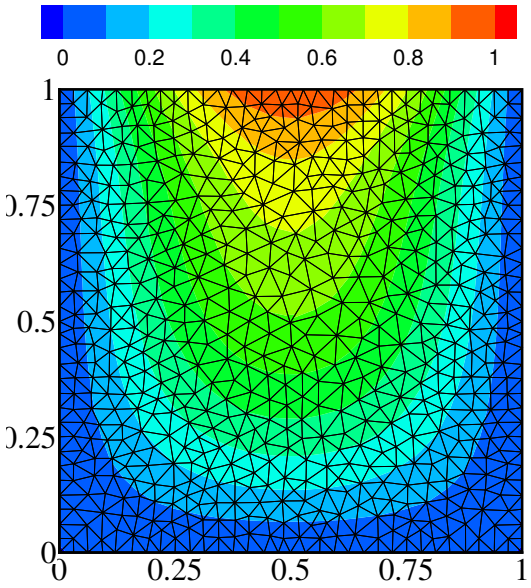
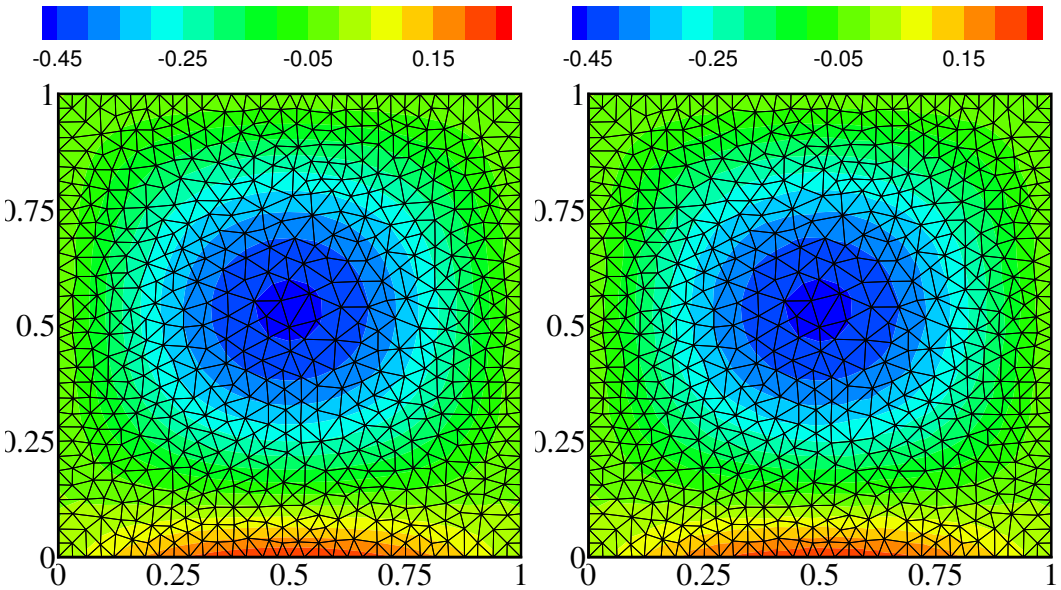


Figure 5.3: Exact solution of the Poisson problem



(a) Continuous adjoint

(b) Discrete adjoint

Figure 5.4: Adjoint solutions for Poisson problem

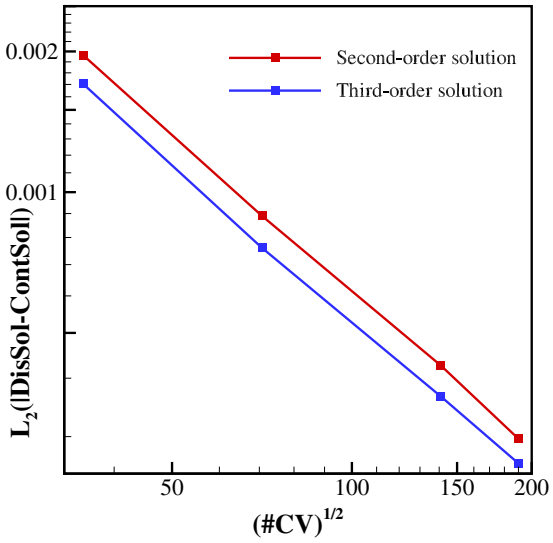


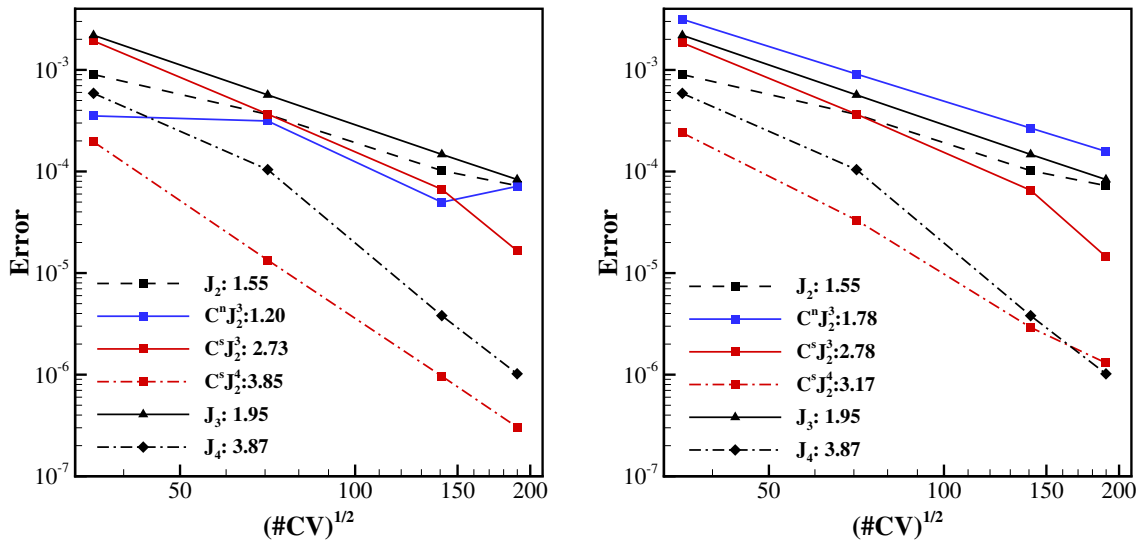
Figure 5.5: The L_2-norm of the difference between the discrete and continuous adjoint solutions for Poisson

output functional is defined as:

$$\begin{aligned} J &= (U,g)_D + (CU,h)_{\partial D_D} = (U,g)_D + (\hat{n}\cdot\nabla U,h)_{\partial D_D} \\ &= \int_0^1\int_0^1 U_{exact}\times\frac{\pi^5}{2}x\,(1-x)\,y\,(1-y)\,dxdy + \\ &\quad \int_0^1 -\pi\sin(\pi x)\left(\frac{\pi}{8}+\frac{1}{\sinh(\pi)}\right)x\,(1-x)\,dx \\ &= 1.832245378 - 0.19424836 = 1.637997017 \end{aligned}$$

The continuous and discrete adjoint solutions are depicted in Figure 5.4 and as indicated the boundary condition is non-zero at the bottom wall and zero elsewhere. For the Poisson problem, the difference between the continuous and discrete adjoint solution distributions is less than the advection problem, at least qualitatively. Figure 5.5 shows the L_2-norm of the difference between the continuous and discrete adjoint solution for both second order and third order solutions. By decreasing the mesh size, the difference between the discrete and continuous adjoint solutions become smaller.

We correct the functional using these adjoint solutions; the results are shown in Figure 5.6. Similar to the advection problem, comparing the functional based on the second-order solution with the corrected functional based on the discrete solution, $C^n J_2^3$, shows poor performance of the corrected functional for both continuous and discrete ad-



(a) Continuous adjoint

(b) Discrete adjoint

Figure 5.6: Convergence history for Poisson

joint solutions. On the other hand, $C^s J_2^3$ has smaller error compared to the second-order functional and it converges cubically. A significant improvement — fourth-order convergence — is obtained for $C^s J_2^4$ based on continuous adjoint for which the magnitude of the error is even smaller than the fourth-order solution. Using the discrete adjoint, cubic convergence is obtained with comparable error magnitude to the fourth-order functional. For the Poisson problem, we are comparing the error to the fourth-order functional instead of the third-order one since the convergence of the third-order functional is only quadratic. This is the behavior of diffusive fluxes for third-order finite volume unstructured schemes and as the third-order functional is not third-order, as shown in Figure 5.6, correcting it with fourth-order flux integration is not helpful either.

Summarizing, for Poisson, significant improvement is obtained by correcting the functional by the truncation error based on the interpolated solution, when either the continuous or discrete adjoint is used, particularly for $C^s J_2^4$, although the improvement is more significant when the continuous adjoint is used.

5.6 Euler Equations

In this part, two different test cases are considered; supersonic flow with an exact solution and subsonic flow with the solution on a very fine mesh as the comparison solution.

5.6.1 Supersonic Flow

The supersonic vortex is an Euler problem with an exact solution for which the exact functional value can be calculated. The exact solution is shown in Chapter 2, Figure 2.11. For the x-momentum component, the convergence results are shown in Figure 5.7 comparing the discretization error for second order accurate solution and the exact truncation error. As mentioned earlier, the magnitude of the truncation error for the unstructured mesh is larger than the discretization error and the convergence rate is slower. The estimation of the truncation error using the discrete p-TE method is also compared to the continuous p-truncation error and the truncation error magnitude is smaller for the interpolated estimate.

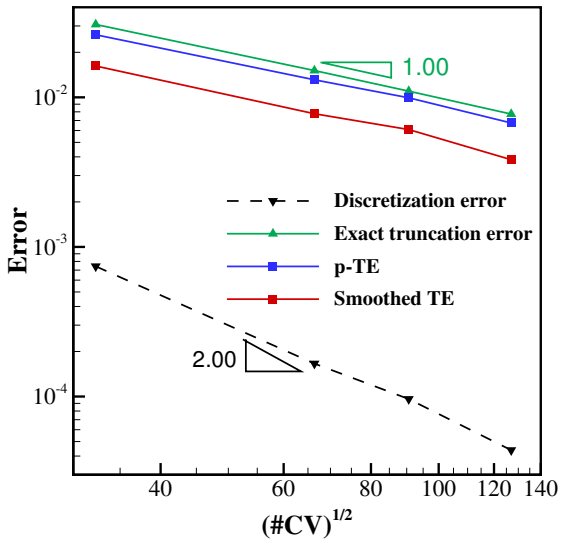
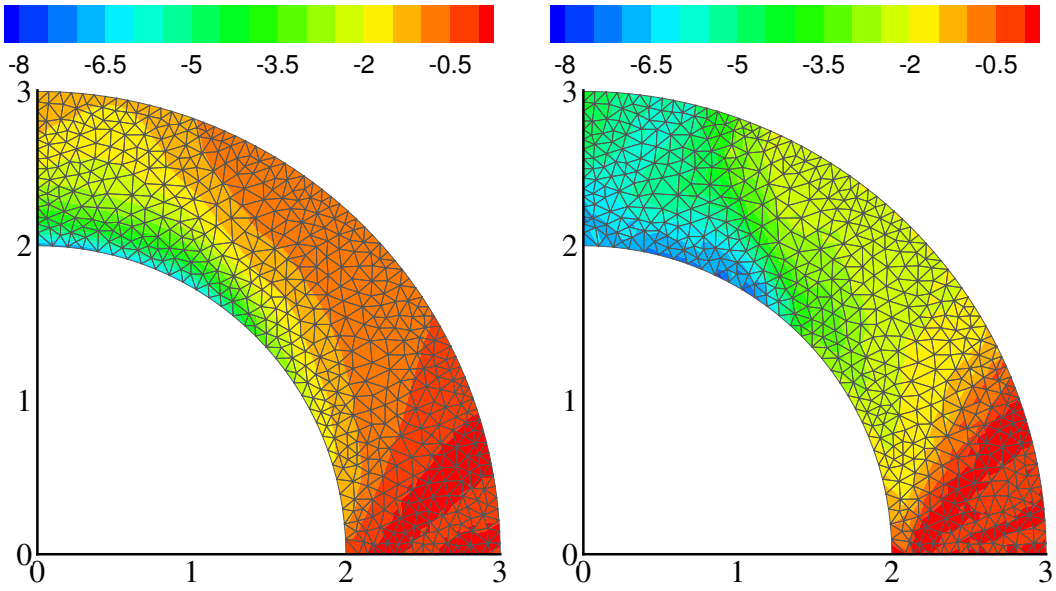


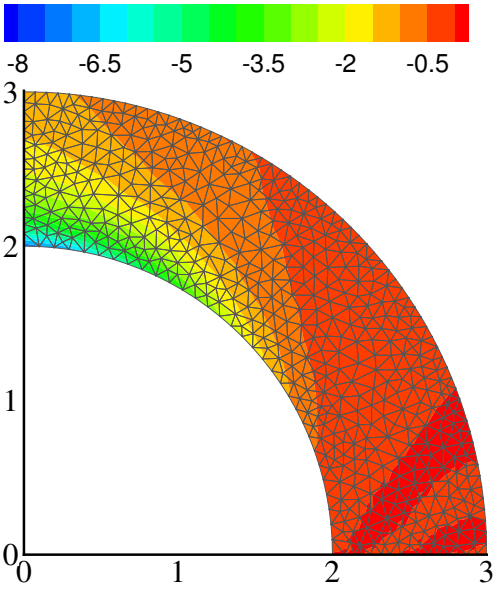
Figure 5.7: Convergence results for the truncation error estimates, supersonic vortex

Considering drag or lift on the inner wall of the supersonic vortex and using Eq. 5.25 for calculating the functional based on the exact pressure distribution, the exact

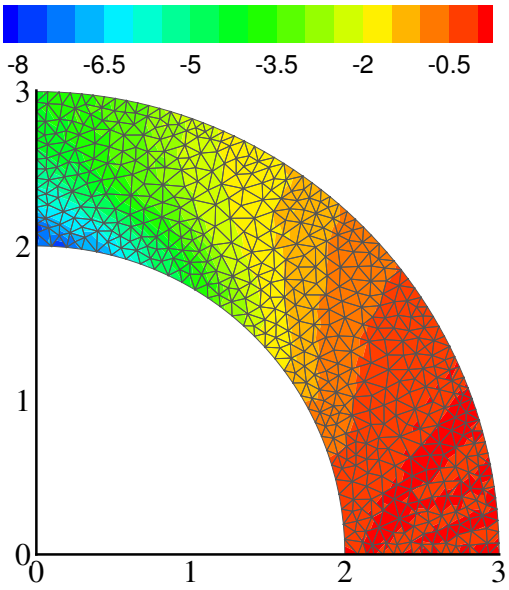


(a) Continuous adjoint for drag functional

(b) Discrete adjoint for drag functional



(c) Continuous adjoint for lift functional



(d) Discrete adjoint for lift functional

Figure 5.8: $x-$momentum adjoint solutions for supersonic vortex

value for the functional is obtained as:

$$\begin{aligned} J &= \int_{inner\,wall} p\hat{n} \cdot \vec{\psi} ds \\ &= \frac{1}{\gamma} \int_{0}^{\frac{\pi}{2}} R_i \hat{n} \cdot \vec{\psi} d\theta = 1.42857 \end{aligned} \tag{5.39}$$

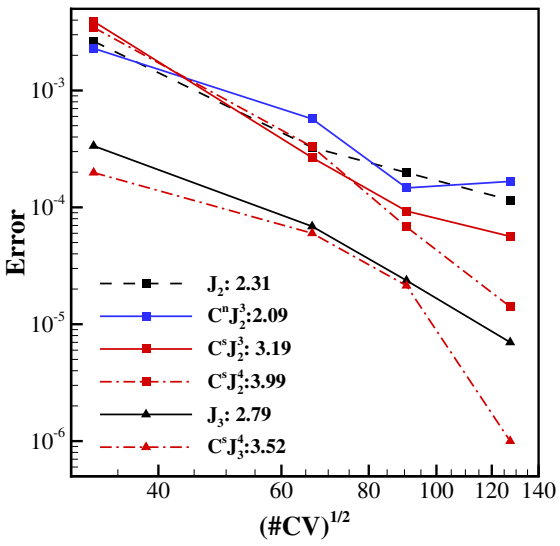
It should be noted that the same magnitude is obtained for both drag and lift.

The continuous and discrete adjoint solutions based on drag and lift functionals on the lower wall are depicted in Figure 5.8. Similar to the advection test case, the continuous adjoint solution distribution is smoother than the discrete adjoint. While both functionals show high sensitivities near the wall, the drag functional adjoint shows high sensitivities along a longer extent than the lift functional due to the need to accurately convect information to the outflow region, where a large amount of the drag force is produced.

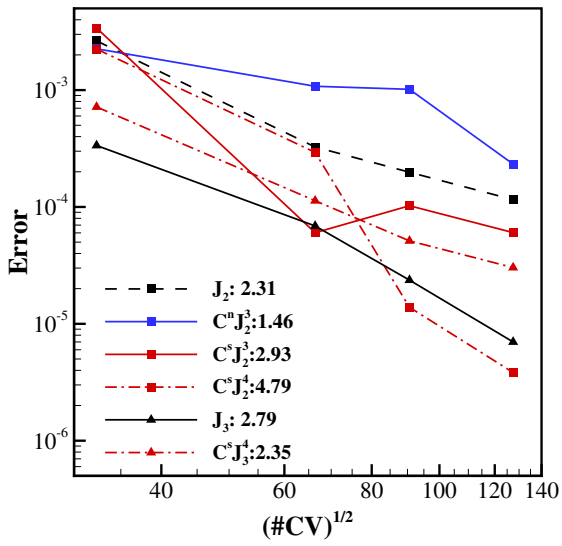
Functional correction based on these adjoint solutions is applied and the results are shown in Figure 5.9. If the discrete p-truncation error is used for functional correction, $C^n J_2^3$, the magnitude of the error in the functional is more than the second order functional and the convergence is approximately quadratic for both drag and lift functionals using both continuous and discrete adjoint solutions. The corrected functional based on the interpolated solution, $C^s J_2^3$, has smaller error compared to the second-order functional and converges cubically. Fourth-order convergence is obtained for $C^s J_2^4$ based on the continuous adjoint for both functionals; however, using the discrete adjoint for $C^s J_2^4$, lift and drag have different behaviors. Although the convergence for drag is even better than quartic, the convergence rate for lift is only 2.54. Generally, using the continuous adjoint for functional corrections provides us with more consistent results, as it is less sensitive to the errors arising from discretization.

If the third-order solution is used for calculating and correcting the functional, the same convergence behavior is obtained using the continuous and discrete adjoint solutions; investigation of convergence behavior for $C^s J_3^4$ using the continuous adjoint shows the significant improvement of correction by using the interpolated solution. The convergence rate is fourth-order and the magnitude of the error is less than the third-order solution. However, using the discrete adjoint for correction is not helpful for $C^s J_3^4$, either in terms of convergence rate, or in terms of the error magnitude.

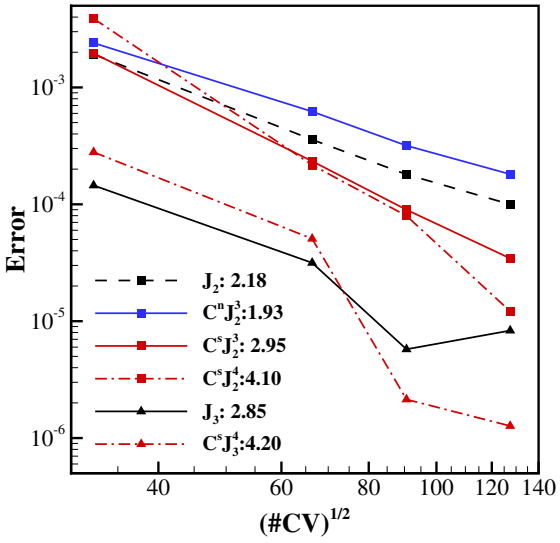
Consequently, as a general conclusion, correcting the functional based on the dis-



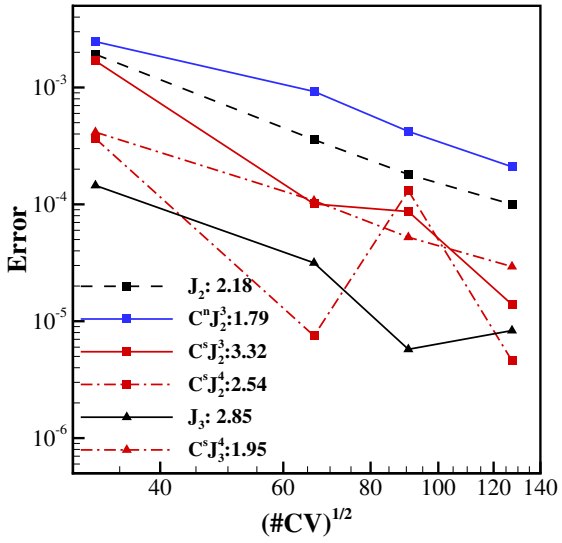
(a) Continuous adjoint for drag functional



(b) Discrete adjoint for drag functional



(c) Continuous adjoint for lift functional



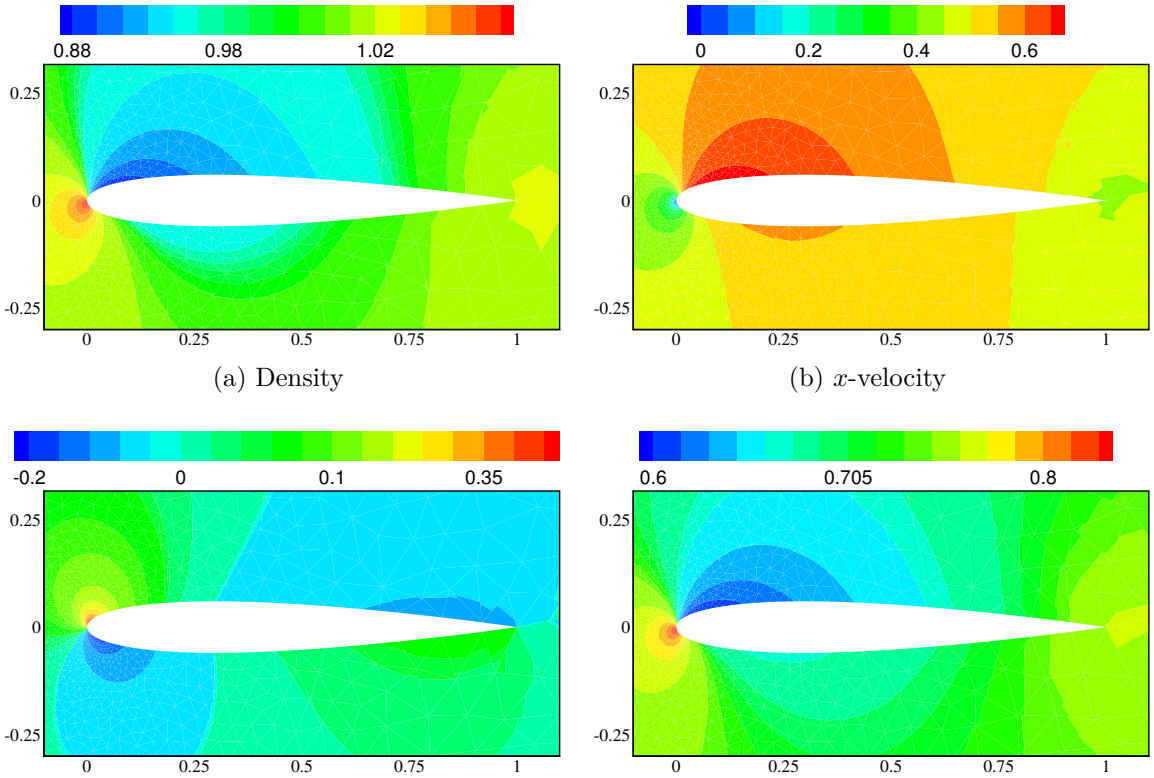
(d) Discrete adjoint for lift functional

Figure 5.9: Convergence history for the functional values for the supersonic vortex

crete solution is not helpful at all and interpolating the solution and computing the correction term based on the interpolated solution is capable of improving the functional significantly. More accurate results with the expected convergence behavior is obtained if the continuous adjoint solution is used for correction.

5.6.2 Subsonic Flow

The capability of our interpolating scheme in correcting the functional for an Euler problem with an exact solution was shown for the supersonic vortex. In this section, output functionals of drag and lift are calculated and corrected for the subsonic inviscid flow around NACA 0012 with $Ma_\infty = 0.5$ and angle of attack $\alpha = 2°$. Since this problem does not have an exact solution, the solution on a very fine mesh is considered as the exact solution. The capability of our solver in approximating the drag and lift



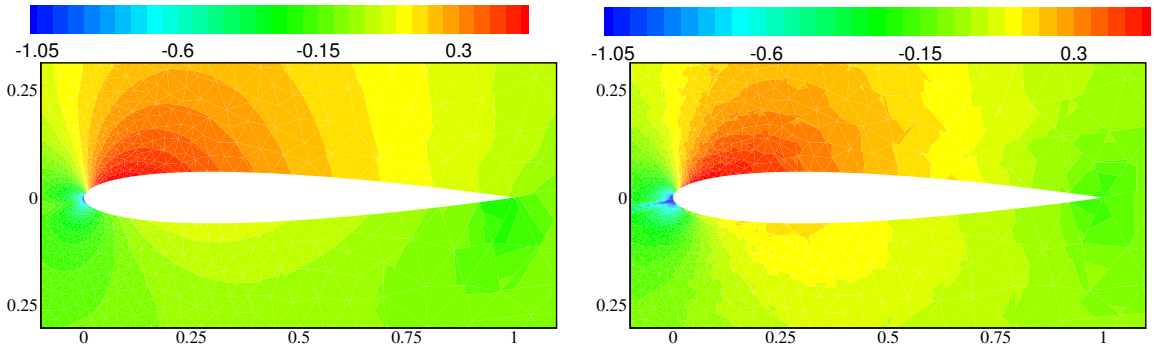
(a) Density (b) x-velocity

(c) y-velocity (d) pressure

Figure 5.10: Solution distribution for Euler, NACA-0012, $Ma_\infty = 0.5$, $\alpha = 2°$

for inviscid flow around NACA 0012 with $Ma_\infty = 0.5$ and $\alpha = 2°$ was shown in the First International Higher-Order Workshop [1]. The solution is depicted in Figure 5.10

and the farfield boundary is a circle of radius 500 chords centered at the leading edge of the the airfoil and consequently, the errors arising from boundary placement can be neglected. Theoretically, the exact drag value for this case should be zero, however farfield effects keep it from being zero numerically. The second-order solution on a mesh with characteristic size nine times smaller in area than the finest mesh for which correction is applied is considered as the exact solution to the problem.



(a) Continuous adjoint for drag functional

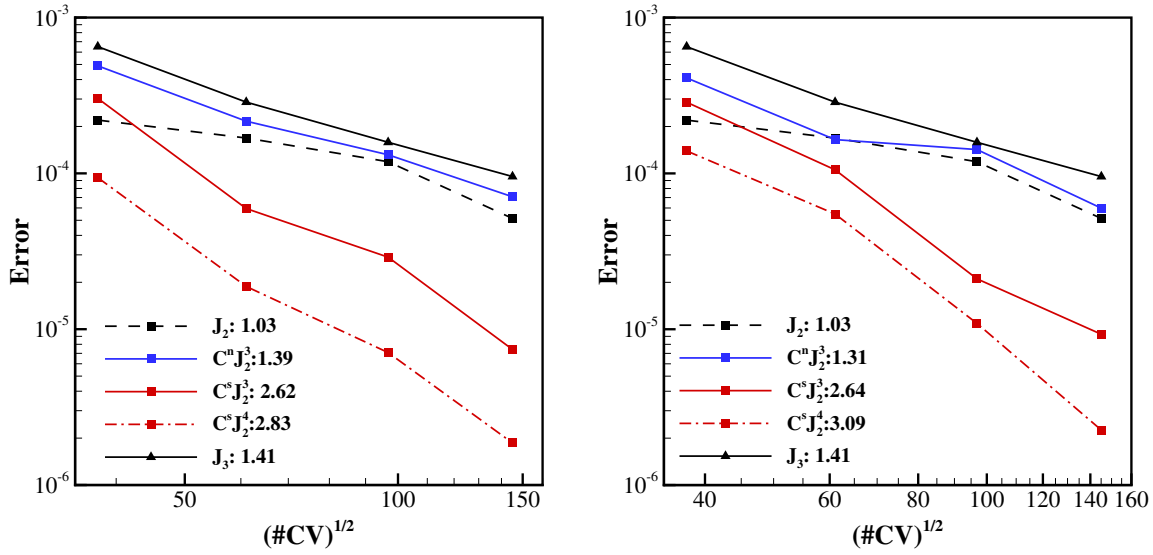
(b) Discrete adjoint for drag functional

Figure 5.11: x-velocity adjoint solutions for Euler, NACA-0012, $Ma_\infty = 0.5$, $\alpha = 2^\circ$

The adjoint solution distributions based on drag functional are shown in Figure 5.11 for both continuous and discrete adjoints and not surprisingly, the continuous adjoint has a smoother distribution.

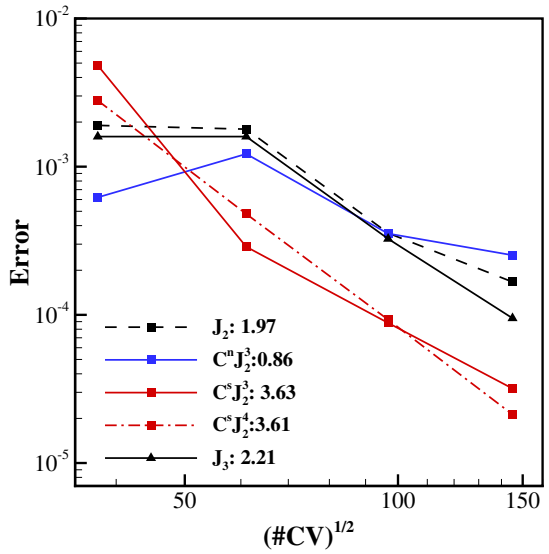
Functional correction based on these adjoint solutions is applied and the results are summarized in Figure 5.12. Again, using the discrete p-truncation error for correcting the functional is not helpful. For the drag functional, the error of the corrected functional based on discrete p-truncation error is more than the second order non-corrected solution. For the lift functional using the discrete adjoint solution, the behavior is the same; however, using the continuous adjoint solution, the magnitude of the error is less but the convergence rate is only 0.86.

Similar to the other cases tested so far, comparing the functional based on the second-order solution with the corrected functional based on the interpolated solution, $C^s J_2^3$ and $C^s J_2^4$, shows that the latter has smaller error and faster convergence rate. For the drag functional, although the non-corrected solution is first-order, the corrected solution based on the continuous p-truncation error converges much faster for both continuous and discrete adjoint solutions. For lift as the output functional, third-order reconstruction of the solution seems to provide us with significant cancellation when

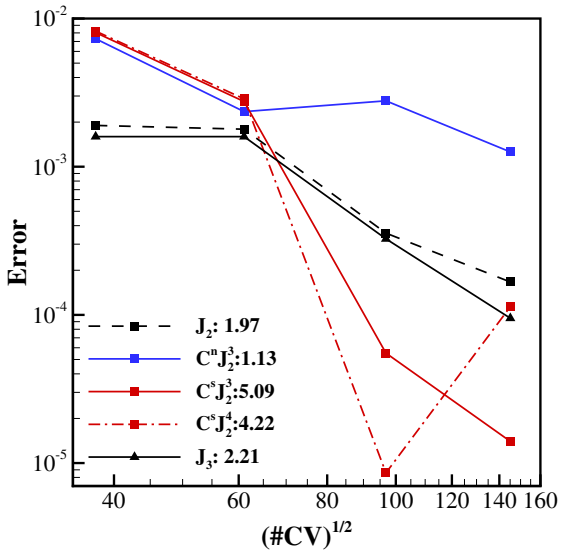


(a) Continuous adjoint for drag functional

(b) Discrete adjoint for drag functional



(c) Continuous adjoint for lift functional



(d) Discrete adjoint for lift functional

Figure 5.12: Convergence history for the functional values for Euler, NACA-0012, $Ma = 0.5$, $\alpha = 2^\circ$

combined with either the continuous or discrete adjoints, but similar to the previous test cases, more consistent results are obtained for $C^s J_2^4$ when the continuous adjoint solution is used. The magnitude of the error is much smaller than the third-order solution for both drag and lift.

5.7 Navier-Stokes Equations

In this section, two different test cases are considered; supersonic flow with a manufactured solution and subsonic flow with the solution on a very fine mesh as the exact solution.

5.7.1 Supersonic Flow

Since there are not any commonly-used Navier-Stokes problems with exact solutions in the literature, we use a manufactured solution [62] to verify the performance of our interpolating scheme for correcting the functional. We use a manufactured solution for both primal [71] and adjoint problems. The general form of the manufactured primal solution is

$$U(x,y) = a_0 + a_x \sin(b_x \pi x + c_x \pi) + a_y \sin(b_y \pi y + c_y \pi) + a_{xy} \sin(b_{xy} \pi y + c_{xy} \pi) \quad (5.40)$$

and the general form of the manufactured adjoint solution is

$$\begin{aligned} Z(x,y) &= a_0 [\sin(\pi(1-x)) \sin(\pi(1-y)) \sin(b_x \pi x + c_x \pi) \\ &\quad \sin(b_y \pi y + c_y \pi) \sin(b_{xy} \pi y + c_{xy} \pi) + (1-x)(1-y)] \end{aligned} \quad (5.41)$$

where the coefficients for four components are as illustrated in Table 5.2 where c is the speed of sound and SI units are used. The inlet Mach number is set as $Ma_i = 2.5$ and the Reynolds number is defined as

$$Re = \frac{\rho_i u_i L}{\mu_i} = 800$$

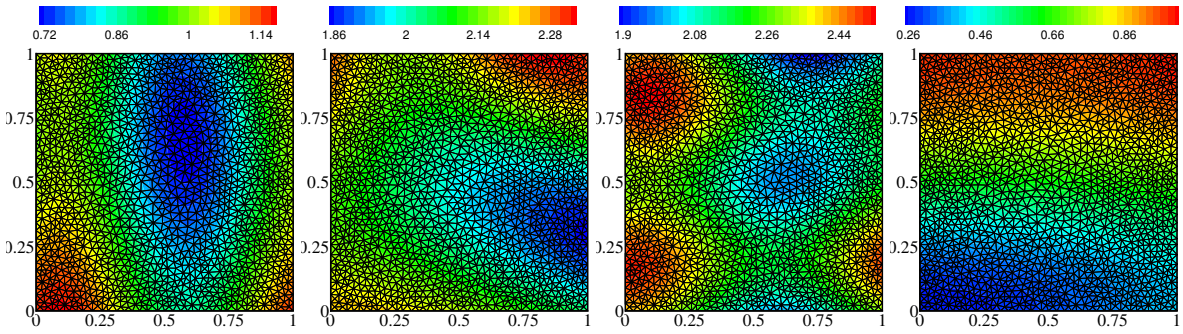
where the characteristic length is $L = 1.0$, $\rho_i = 1.0$ and $\mu_i = 1.0$.

The exact primal solution is shown in Figure 5.13. Since we are using a manufactured solution for both primal and adjoint problems, the functional is directly obtained by the adjoint definition, Eq. 5.9. The source term and boundary conditions of the

	ρ	u	v	P
a_0	1.0	$800/c$	$800/c$	$10^5/c^2$
a_x	0.15	$50/c$	$-75/c$	$-2 \times 10^4/c^2$
b_x	2.0	0.66	1.66	0.5
c_x	0.33	0.5	-0.5	0.5
a_y	-0.1	$-30/c$	$40/c$	$5 \times 10^4/c^2$
b_y	1.0	1.5	3.0	1.0
c_y	-0.2	-0.125	0.0	-0.5
a_{xy}	-0.05	$-60/c$	$60/c$	$-1 \times 10^4/c^2$
b_{xy}	1.0	1.5	1.5	0.75
c_{xy}	0.25	0.125	0.125	-0.25

Table 5.2: Coefficients for manufactured primal and adjoint solution used in Eq. 5.40 and 5.41.

manufactured primal and adjoint solutions are calculated easily and the corresponding weight boundary operators are found by integration by parts, similar to the Dirichlet and Neumann boundary operators obtained for the Poisson problem.

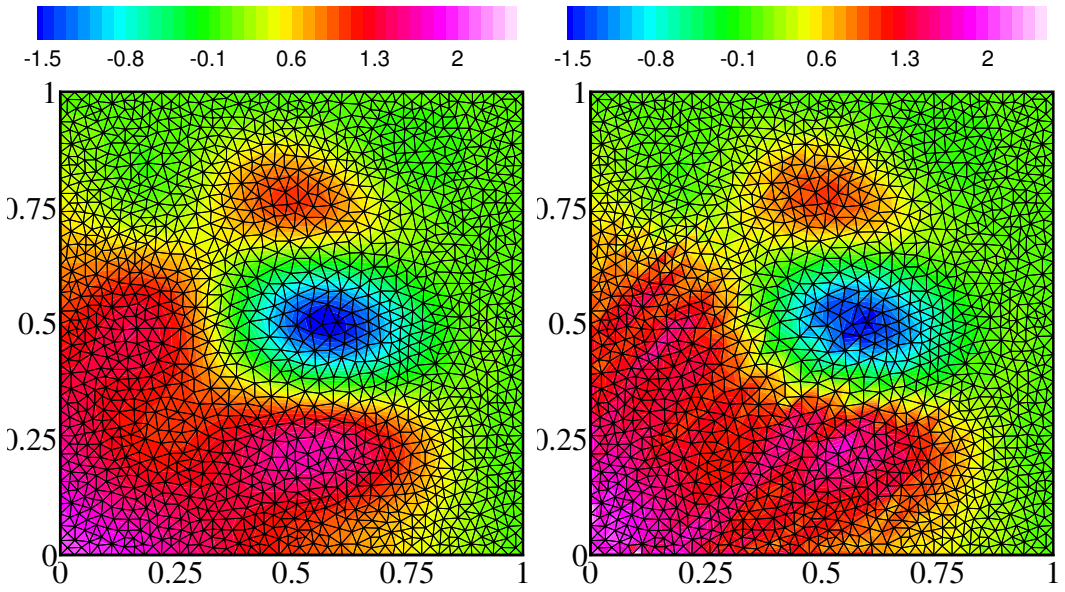


(a) Density (b) x-velocity (c) y-velocity (d) Pressure

Figure 5.13: Exact solution of Navier-Stokes problem with manufactured solution

The $y-$momentum adjoint solution distribution is depicted in Figure 5.14 for both continuous and discrete adjoints. Functional correction based on these adjoint solutions is applied and the results are shown in Figure 5.15. Similar to the Euler problem, by interpolating the solution and computing the truncation error based on the interpolated solution, $C^s J_2^3$, the magnitude of the error compared to the second-order functional becomes smaller and it converges cubically using the continuous adjoint for correction. Correcting the functional by the same adjoint solution, fourth-order convergence is obtained for $C^s J_2^4$. However, using the discrete adjoint for both $C^s J_2^3$ and $C^s J_2^4$ gives a

corrected functional with smaller error but only quadratic convergence. Alternatively, functional correction may be based on the third-order solution; investigation of convergence behavior for $C^s J_3^4$ shows improvement if correction is applied by the interpolated solution and the continuous adjoint. The convergence rate is almost fourth-order and the error is smaller than the error for the third-order solution. However, by using the discrete adjoint for correction, the magnitude of the error is less than the uncorrected functional and convergence is cubic. Similar to the conclusion for the previous test case, using continuous adjoint for correction is able to improve the functional more and provide us a faster convergence as well.

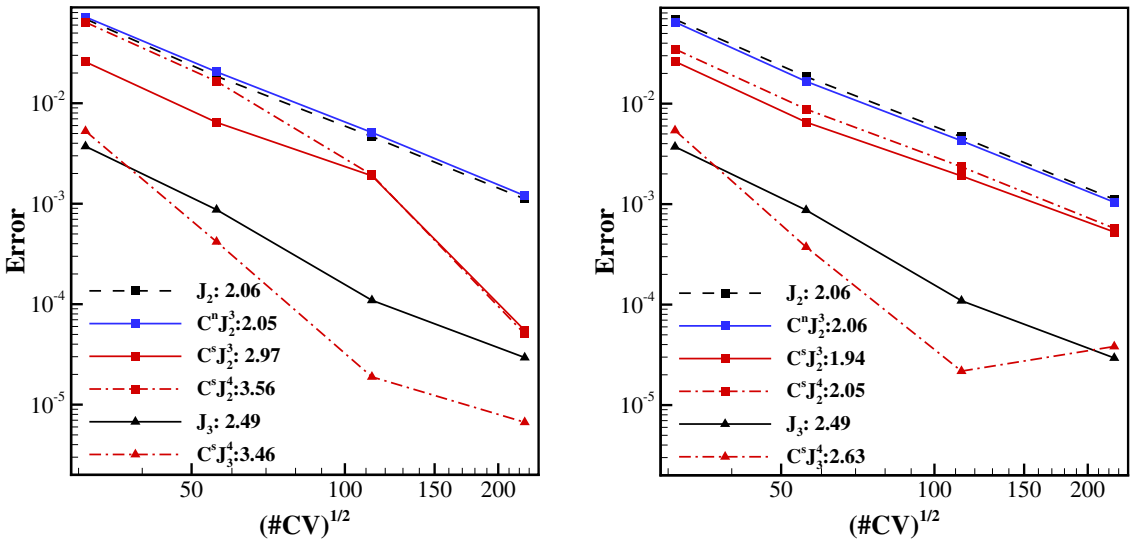


(a) Continuous adjoint

(b) Discrete adjoint

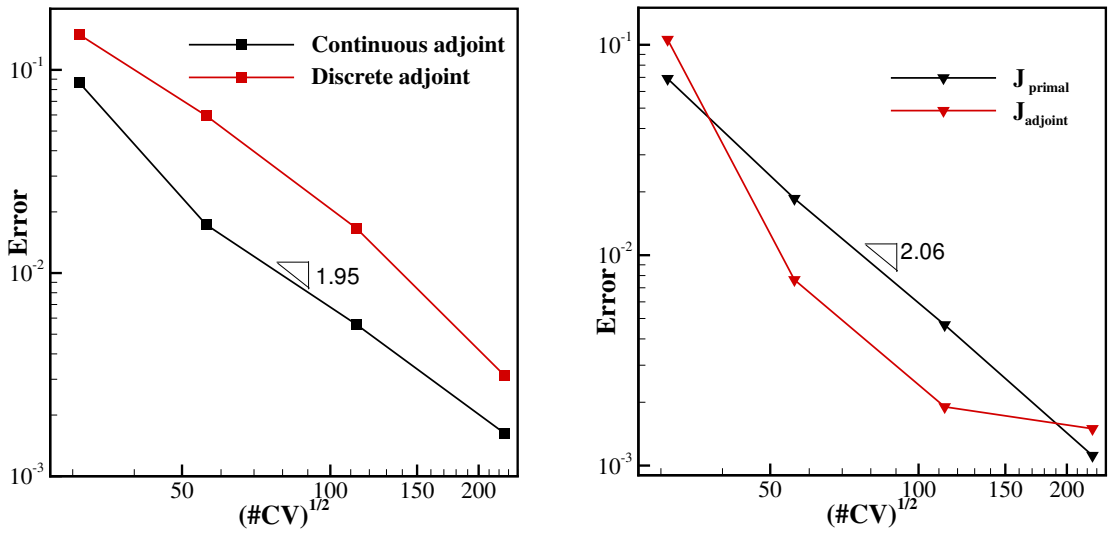
Figure 5.14: Adjoint solutions for Navier-Stokes problem with manufactured solution, $y-$momentum

As the manufactured solution is set for the adjoint problem, the magnitude of the error of the adjoint solution can be calculated. Figure 5.16a depicts the error for both discrete and continuous solutions. Not surprisingly, the magnitude of the error in calculating the continuous adjoint is less than the discrete adjoint and the difference becomes smaller by decreasing the mesh size; both converge quadratically. Adjoint consistency is verified for this test case, as it is a system of equations with both convective and diffusive terms. Adjoint consistency for the discrete adjoint solution for this test case is depicted in Figure 5.16b which shows that almost the same functional value is obtained either



(a) Continuous adjoint (b) Discrete adjoint

Figure 5.15: Convergence history for the functional values for Navier-Stokes problem with manufactured solution



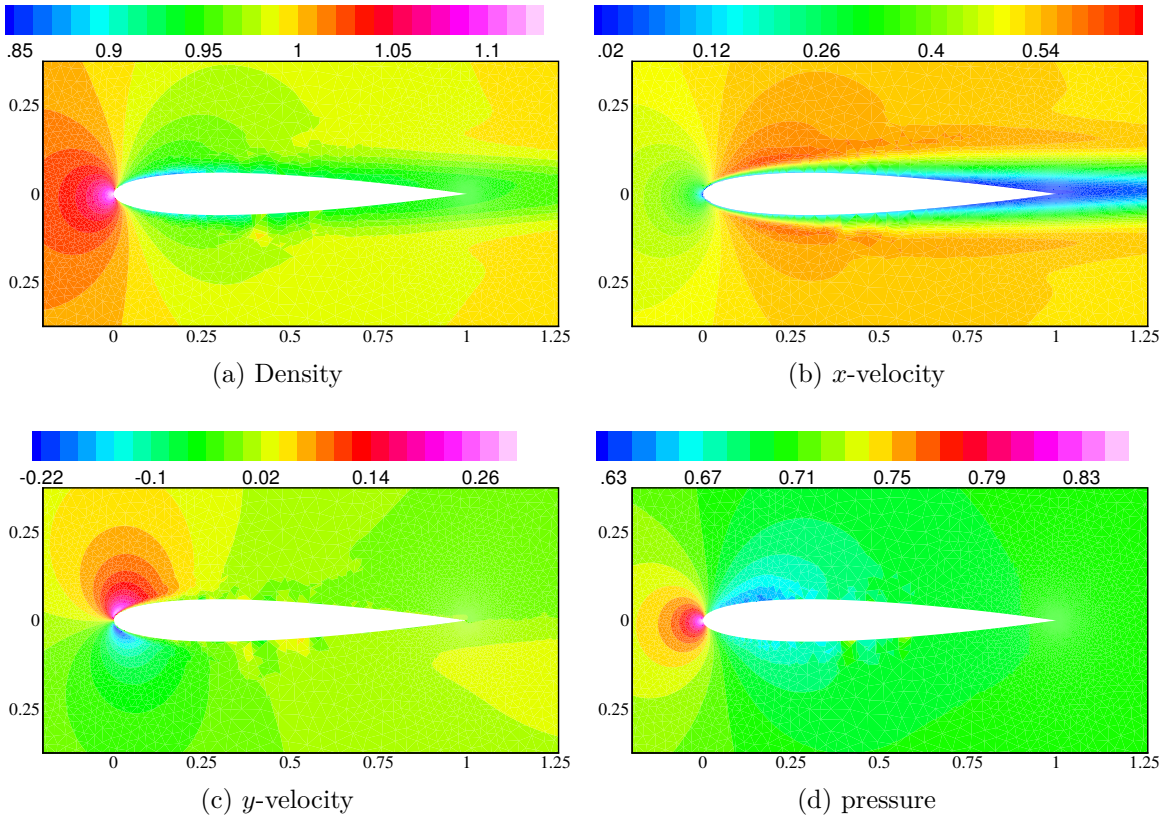
(a) Continuous and discrete adjoint solutions (b) Adjoint Consistency

Figure 5.16: Adjoint consistency for Navier-Stokes

based on the primal solution or the adjoint solution and the difference decreases with mesh refinement, implying that the discretization is asymptotically adjoint consistent.

5.7.2 Subsonic Flow

In this section, drag is calculated and corrected for the subsonic viscous flow around NACA 0012 with $Ma_\infty = 0.5$, angle of attack $\alpha = 1°$ and Reynolds number $Re = 5000$. The farfield boundary is a circle of radius 100 chords centered at the leading edge of the the airfoil. Since this problem does not have an exact solution, the drag calculated using the second-order solution on a mesh with characteristic size thirteen times smaller in area than the finest mesh is considered as the exact solution.

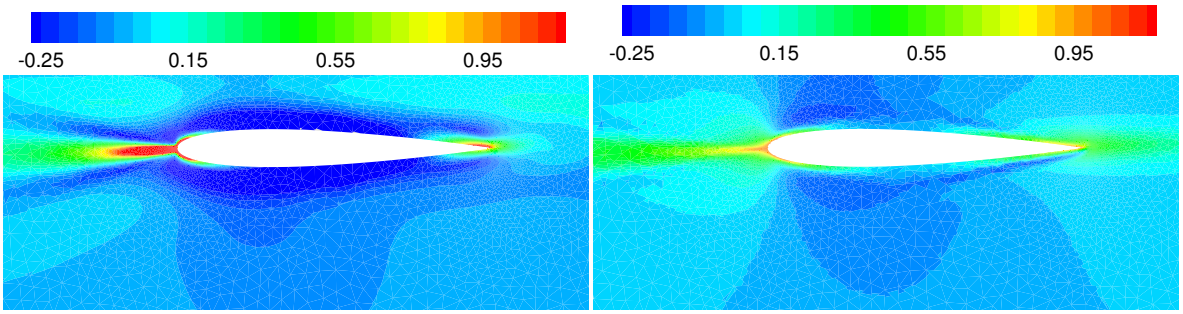


(a) Density (b) x-velocity

(c) y-velocity (d) pressure

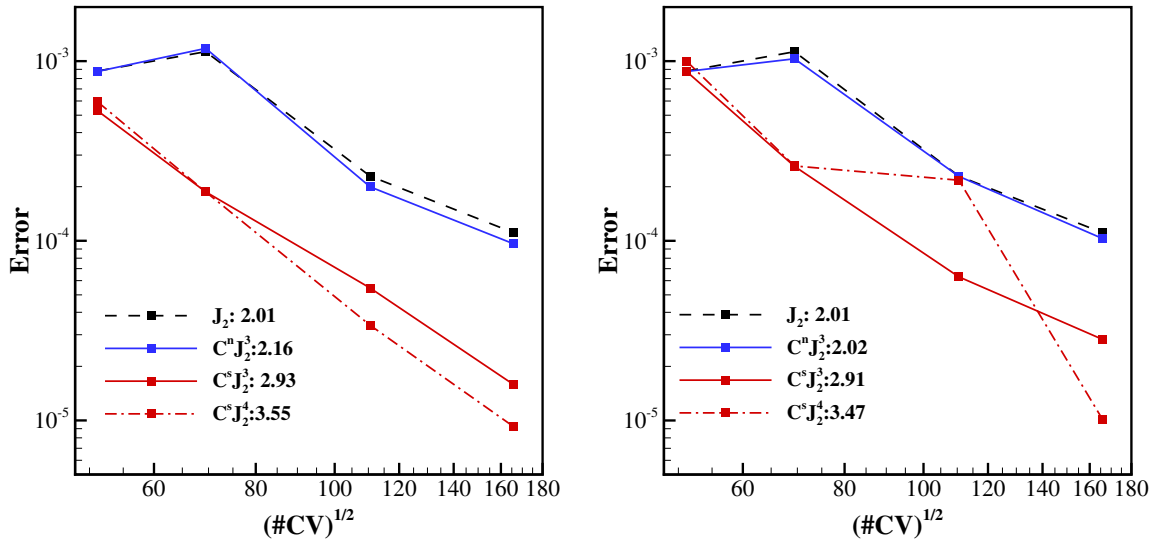
Figure 5.17: Solution distribution for Navier-Stokes, NACA-0012, $Ma_\infty = 0.5$, $\alpha = 1°$, $Re = 5000$

Four components of the solution are depicted in Figure 5.17. The adjoint solution distributions based on the drag functional are shown in Figure 5.18 for both continuous and discrete adjoints.



(a) Continuous adjoint (b) Discrete adjoint

Figure 5.18: x-momentum component of the adjoint solutions for drag functional, Navier-Stokes NACA-0012, $Ma_\infty = 0.5$, $\alpha = 1°$, $Re = 5000$



(a) Continuous adjoint (b) Discrete adjoint

Figure 5.19: Convergence history for Navier-Stokes, NACA 0012 for drag functional, $Ma_\infty = 0.5$, $\alpha = 1°$, $Re = 5000$

Functional correction based on these adjoint solutions is applied and the results are summarized in Figure 5.19. Correcting the drag based on the discrete p-truncation error, $C^n J_2^3$, is not helpful compared to the original second order solution. Correcting the functional by the continuous p-truncation error is able to improve the functional. In particular, cubic and approximately quartic convergence is obtained by correcting the functional using the continuous p-truncation error calculated by third and fourth order flux integration, respectively. Similar to other test cases, using the continuous adjoint solution provides more consistent results.

5.8 Remarks on Computational Costs and Discussion on Results

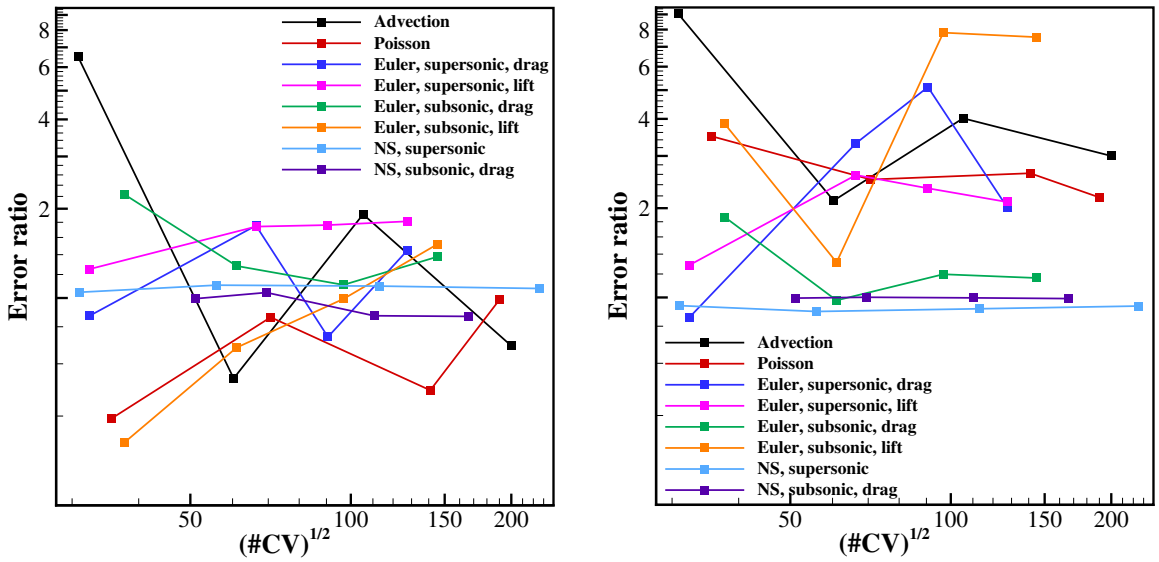
The point that needs to be considered about computational cost is the time comparison between continuous and discrete adjoint solutions. For the discrete adjoint, the computational cost includes finding the transpose of the Jacobian and the derivative of the functional with respect to the solution and then solving the system of equations once for which the computational cost is of the order of one linear iteration of the solution. On the other hand, for the continuous adjoint, the computational time is at the same order of computational time required for the primal problem. Hence, the total time for correcting the second-order functional using our method with the continuous solution adjoint is twice as large as solving the primal problem, while the computational time for correction using the discrete adjoint is the equal to the time for solving the primal problem plus one non-linear iteration.

The significant improvement in functional correction is obtained just by interpolating the primal solution and finding the truncation error based on this interpolated solution and then multiplying the truncation error by the adjoint solution of the same order as the primal solution. It is not necessary to interpolate the adjoint solution to obtain significant improvement in convergence rates.

A summary of the results for different test cases described is shown in this section. For each case, the error in calculating the functional based on the second-order solution is considered as the reference value and the ratio of the error in the corrected functional to the non-corrected second-order functional is compared for different test cases and shown in Figure 5.20. The nomenclature is the same as described earlier and the comparison is depicted for correcting based on the discrete and continuous p-truncation

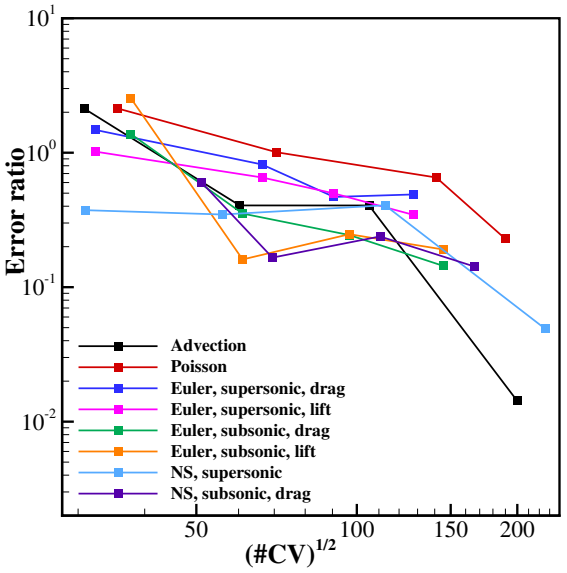
error for both continuous and discrete adjoint solutions for all test cases described in this chapter. Correcting based on the discrete p-truncation error is not helpful and in many cases, the ratio of the error is even more than one. The same behavior exists for both discrete and continuous adjoint solution.

Using the continuous p-truncation error, on the other hand, the error of the corrected functional is smaller than the original non-corrected one and using the continuous adjoint solution, it becomes smaller with mesh refinement. As mentioned earlier, continuous adjoint solution provides more consistent results. Moreover, the ratio of the corrected functional using the fourth-order flux integral with respect to the non-corrected second-order functional is less than the corresponding ratio for the third-order flux integration. Note that the same scale is used for the plots using the continuous p-truncation error.

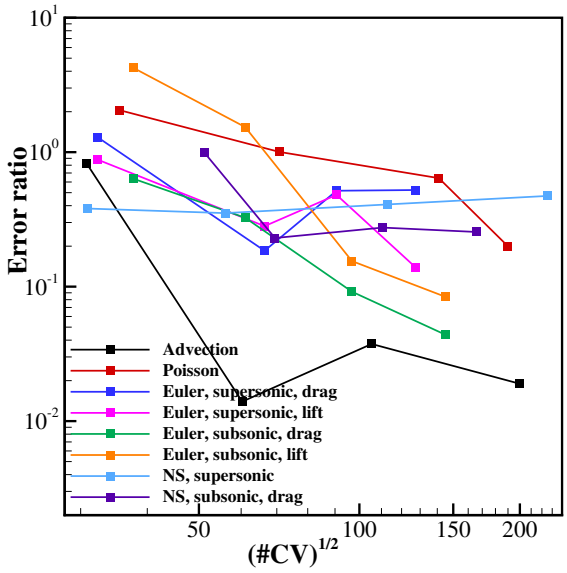


(a) $C^n J_2^3$ using the continuous adjoint solution

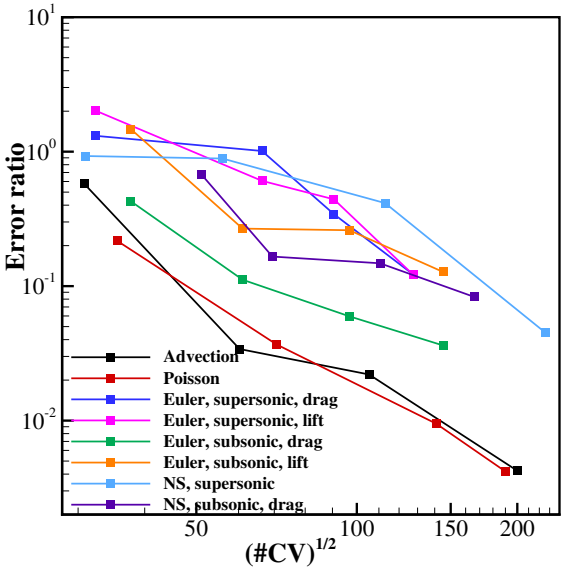
(b) $C^n J_2^3$ using the discrete adjoint solution



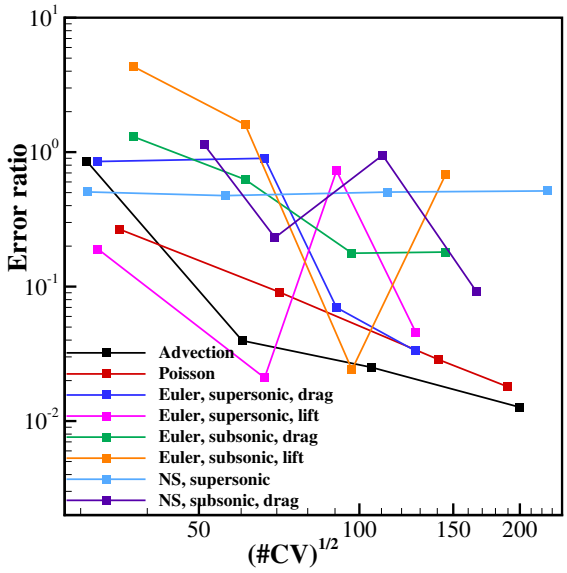
(c) $C^s J_2^3$ using the continuous adjoint solution



(d) $C^s J_2^3$ using the discrete adjoint solution



(e) $C^s J_2^4$ using the continuous adjoint solution



(f) $C^s J_2^4$ using the discrete adjoint solution

Figure 5.20: The error ratio for different correction terms for all test cases based on second-order functional

Chapter 6

Mesh Adaptation

The equidistribution of error, or of weighted error, is the principle of all mesh adaptation strategies [42]. The objective of equidistribution is defined as finding a mesh such that the integral of the weight function defined on a domain takes a given constant value over each element or equivalently, such that the weight function is equidistributed over the chosen mesh. For mesh adaptation, we need to set a length scale for each vertex and the length scale is calculated based on an adaptation indicator.

Consider residual-based adaptation in which the truncation error, τ is chosen as the adaptation indicator. The average truncation error, $\overline{\tau}_h$ is defined as:

$$\overline{\tau}_h = \frac{\sum_{i=1}^{N} A_i \tau_i}{\sum_{i=1}^{N} A_i} \tag{6.1}$$

where τ_i is the truncation error on each cell i, A_i is the area of the cell and the N is the total number of cells. The target area for each cell is set as:

$$A_i^* = \frac{\overline{\tau}_h}{2\tau_i} A_i \tag{6.2}$$

which reduces area by a factor of 2, on average. The length scale for regenerating the unstructured mesh using a Delaunay mesh algorithm is defined as:

$$l_i^* = C_l \sqrt{A_i^*} \tag{6.3}$$

where $C_l = \sqrt{3}$. For isotropic meshes, the length scale assigned to each vertex can be calculated simply by averaging the length scales associated with all cells sharing that vertex. For generating the initial mesh and the adapted meshes, the GRUMMP meshing library is used [63]. In Eq. 6.1 and 6.2, the truncation error may be replaced by the truncation error weighted by the adjoint solution or any other adaptation indicator.

As mentioned in Chapter 1, different adaptation indicators may be exploited. Three

different adaptation indicators, residual-based, adjoint-based and adaptation based on higher-order error in the correction term are described in this chapter.

As test cases, we will start with a Poisson problem with exact solution, so the exact error in calculating the functional of interest on the adapted mesh can be calculated. The flow test cases being investigated here are the subsonic inviscid flow around a NACA 0012 airfoil and a multi-element airfoil, and subsonic viscous flow and transonic inviscid flow around a NACA 0012 airfoil. Similar to the real flow test cases described in the previous chapter, we will use the solution on a very fine mesh for test cases with unknown exact solution.

6.1 Residual-based Mesh Adaptation

A typical residual-based mesh adaptation uses an estimate of the truncation error as the adaptation indicator. The higher-order flux integral based on the lower-order solution or the discrete p-TE estimate is a common indicator for residual-based mesh adaptation. However, it may be replaced by the continuous p-TE estimate we have developed using the C^1 interpolation of the solution. We will compare the effectiveness of the original discrete p-TE estimate and the the continuous p-TE estimate as the adaptation indicator in this chapter.

6.2 Adjoint-based Mesh Adaptation by Correction Term

The truncation error weighted by the adjoint solution, or adjoint-based adaptation, is based on a method developed by Pierce and Giles for estimating error in the functional [73]. For a given smoothly interpolated primal, U_h, and adjoint, Z_h, solutions, the output functional is obtained as:

$$\begin{aligned} J &= (g,U)_D + (h,CU)_{\partial D} \\ &= (L^*Z_h,U_h)_D + (L^*Z_h,U-U_h)_D + (g-L^*Z_h,U-U_h)_D + (g-L^*Z_h,U_h)_D + \\ &\quad (B^*Z_h,CU_h)_{\partial D} + (B^*Z_h,C\,(U-U_h))_{\partial D} + \\ &\quad (h-B^*Z_h,C\,(U-U_h))_{\partial D} + (h-B^*Z_h,CU_h)_{\partial D} \end{aligned} \tag{6.4}$$

By using integration by parts, we can re-write the above equation as

$$\begin{aligned} J &= (L^* Z_h, U_h)_D + (B^* Z_h, C U_h)_{\partial D} + \\ &\quad (Z_h, L(U - U_h))_D + (C^* Z_h, B(U - U_h))_{\partial D} + \\ &\quad (Z - Z_h, L(U - U_h))_D + (C^*(Z - Z_h), B(U - U_h))_{\partial D} \end{aligned} \tag{6.5}$$

The two terms in the first line are the influence of bulk and boundary integrals in the discrete functional, the terms in the second line represent computable adjoint error estimates and the last two terms in the third line are the higher order remaining errors.

Neglecting the last two higher-order terms, the estimated error in calculating the functional is:

$$\begin{aligned} \delta J &= (Z_h, L(U - U_h))_D + (C^* Z_h, B(U - U_h))_{\partial D} \\ &= (Z_h, f - L U_h)_D + (C^* Z_h, e - B U_h)_{\partial D} \\ &= (Z_h, \tau)_D + (C^* Z_h, e - B U_h)_{\partial D} \end{aligned} \tag{6.6}$$

Therefore, the dot product of the adjoint solution and the truncation error of the primal problem is used as the adaptation indicator for adjoint-based adaptation. For this purpose, there are multiple options: using the original p-TE estimate by higher order flux integration based on the lower order solution or the continuous p-TE estimate as described in previous chapter. Either of these estimates of the truncation error may be weighted by the discrete or continuous adjoint solution.

As described in Chapter 5, the correction term of Eq. 6.6 is the summation over all control volumes of the truncation error calculated by higher-order flux integral based on lower-order solution multiplied by the adjoint solution of the same order as the primal problem

$$\delta J = (Z_h, \tau)_D = \sum_{CV} \left(\oint \vec{F}_p^{p+1} \cdot \hat{n} ds \right) \bar{Z}_p \tag{6.7}$$

in contrast to the finite-element method where the higher-order adjoint solution is required to find the correction term.

6.3 Adjoint-based Mesh Adaptation by Error in the Correction Term

An adaptive strategy may be based on improving the accuracy of the computable error estimates instead of improving the accuracy of the functional directly. The latter could lead in some cases to refining or coarsening in regions where the adjoint solution is not sufficiently resolved and as a consequence, an adaptation indicator which is based on both the primal and adjoint residual errors may result in improvement in the quality of the error estimates. Furthermore, it leads to improvement in the base value of the functional even before correction.

In Eq. 6.6, the higher order error terms have been ignored, but if all terms are considered, we will get:

$$\begin{aligned}
\delta J &= (Z_h, f - LU_h)_D + (C^* Z_h, e - BU_h)_{\partial D} + \\
&\quad (Z - Z_h, L(U - U_h))_D + (C^*(Z - Z_h), e - B(U_h))_{\partial D} \\
&= (Z_h, \tau)_D + (C^* Z_h, e - BU_h)_{\partial D} + \\
&\quad (Z - Z_h, L(U - U_h))_D + (C^*(Z - Z_h), e - B(U_h))_{\partial D}
\end{aligned} \tag{6.8}$$

The first term in the last line is the truncation error of the primal problem weighted by the discretization error of the adjoint solution which is used here as the adaptation indicator. Calculating the discretization error of the adjoint problem is not easy for general problems where the exact adjoint solution is not known; however, this term can be recast as a function of the Jacobian matrix [94]:

$$(Z - Z_h, L(U - U_h))_D = \left(L(Z - Z_h), \left(\frac{\partial R}{\partial U_h} \right)^{-1} L(U - U_h) \right)_D \tag{6.9}$$

where the first term is the truncation error of the adjoint solution. This is evaluated by higher-order flux integration based on the lower-order solution and is already available. Evaluating the second term is as easy as solving a linear system of:

$$X = \left(\frac{\partial R}{\partial U_h} \right)^{-1} L(U - U_h) \Longleftrightarrow \left(\frac{\partial R}{\partial U_h} \right) X = L(U - U_h) \tag{6.10}$$

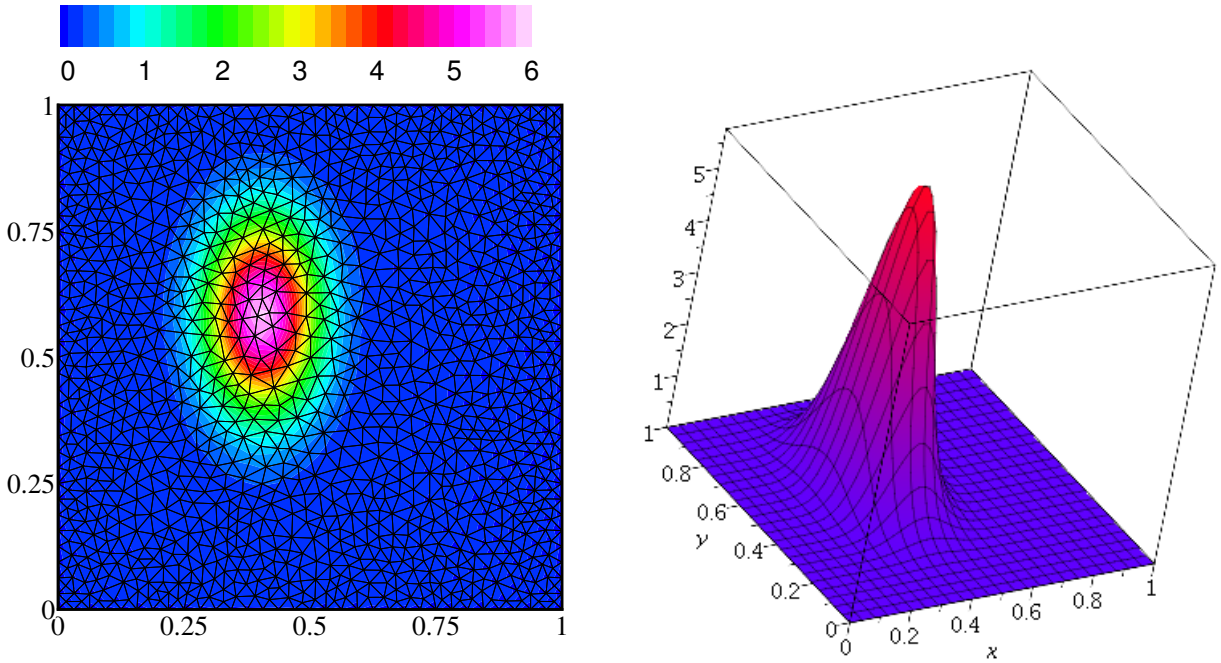
Accordingly, this term can be used as the adaptation indicator capable of reducing the

error in corrected functional.

In the following sections, the effectiveness of these adaptation indicators are studied for several test cases.

6.4 Poisson Equation

The primal problem is defined on a (1×1) square domain with zero boundary conditions such that the solution has a peak value and is almost zero elsewhere. This has been chosen so that the adaptation effect is noticeable.



(a) Solution contour

(b) 3D view

Figure 6.1: Exact solution of the Poisson problem

The exact primal solution is:

$$u_{exact} = 100xy\,(1-x)\,(1-y)\,e^{-81(x-0.4)^2-25(y-0.6)^2} \tag{6.11}$$

The exact solution is shown as a 3D surface in Figure 6.1 to emphasis the peak point. Setting the adjoint source term as $g = \frac{\pi^5}{2}x\,(1-x)\,y\,(1-y)$ with zero boundary condi-

tion, the exact output functional is obtained as:

$$\begin{aligned} J &= (u, g)_D \\ &= \int_0^1 \int_0^1 u_{exact} \times \frac{\pi^5}{2} x\,(1-x)\,y\,(1-y)\,dxdy \\ &= 2.936927271 \end{aligned}$$

We have shown in Chapter 5 that using the continuous p-truncation error weighted by the continuous adjoint solution as the correction term provides the best correction in the output functional. Therefore, we are comparing the results with the best corrected functional on each mesh, using the continuous adjoint solution and the continuous p-truncation error. As the exact functional is known, calculating the error in the functional for each case is easy.

Before showing the results, the nomenclature we are using should be described; UCJ is the corrected functional on the uniformly refined mesh where C's superscript represents whether the correction is based on truncation error using the discrete solution, C^n, or is based on the truncation error using the interpolated solution, C^s. C's subscript represents whether continuous adjoint solution, C_c, or discrete adjoint solution, C_d, is used for obtaining the correction term. Similar to the Chapter 5, J's subscript shows the primal solution order of accuracy and the superscript shows the order of reconstruction and flux integration used as an estimation of the truncation error. The solid lines shown by $A\,()$ are the adapted meshes and the adaptation indicator is defined inside the parenthesis. (τ^n) is residual-based mesh adaptation using the discrete p-truncation error, (τ^s) is residual-based using the continuous p-truncation error based on interpolated solution, (τ^s, D) is the adjoint-based adaptation using continuous p-truncation error weighted by the discrete adjoint solution, (τ^s, C) is the adjoint-based adaptation using the continuous p-truncation error weighted by continuous adjoint solution, and $\left(\tau_p^s, \varepsilon_a\right)$ is the primal truncation error weighted by adjoint discretization error as the adaptation indicator representing adaptation based on higher-order error in correction term.

The convergence results for the functional on the adapted meshes are shown in Figure 6.2. The dashed red line is the corrected functional based on the continuous adjoint and the continuous p-truncation error on uniformly refined meshes and the solid lines are the adapted meshes. The gradient symbol denotes using the truncation error as the adaptation indicator, the square symbol represents adjoint-based adaptation

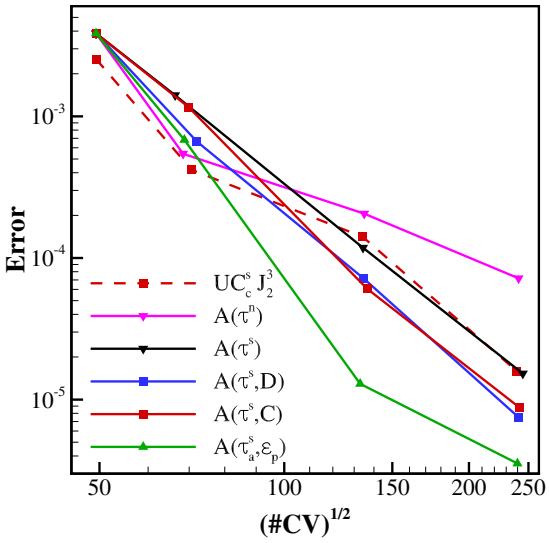
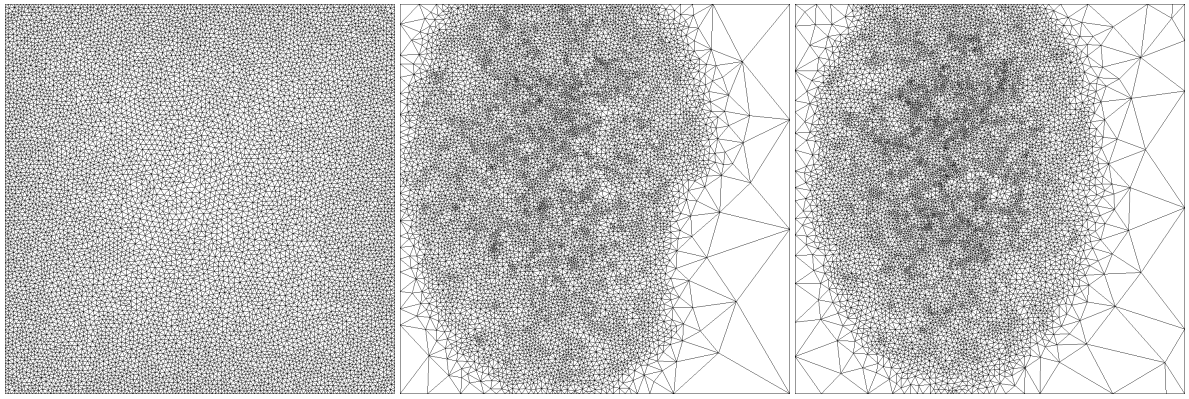


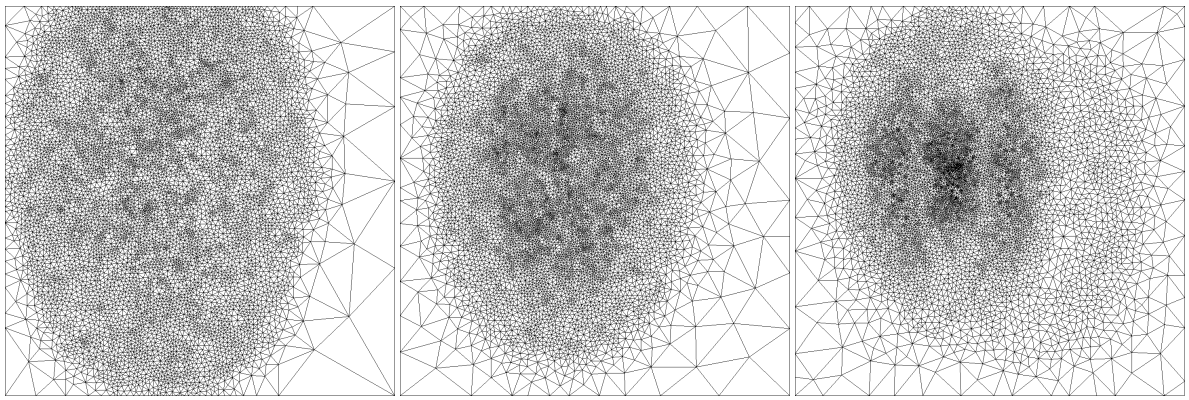
Figure 6.2: Convergence history for Poisson on the adapted meshes

indicator using the correction term based on the continuous p-truncation error, Eq. 6.7 discussed in section 6.2, and the triangle symbol denotes the adjoint-based adaptation indicator using higher-order terms, Eq. 6.9, discussed in section 6.3. It should be noted that the uniformly refined meshes are set such that the number of cells are approximately comparable to number of cells of the adapted meshes.

The magenta line is the adapted mesh based on discrete p-truncation error and as shown, the magnitude of the error is more than the corresponding error on the uniformly refined meshes and the convergence rate is slower. Adaptation based on continuous p-truncation error, however, gives results comparable to the uniformly refined meshes. Using the correction term as the adaptation indicator based on continuous p-truncation error and adjoint solution provides functionals with smaller error and faster convergence. The adjoint solution type does not matter: both the discrete adjoint (the blue line) and the continuous adjoint (the red line) have approximately the same behavior. Furthermore, if the higher-order terms representing the error in calculating the correction term are used for mesh adaptation (the green line) the convergence rate improves significantly and the magnitude of the error is much smaller compared to other adapted meshes. It should be noted that for this case, as the exact primal solution is known, the discretization error of the primal solution is easily computable. Accordingly, instead of using Eq. 6.9, an alternative dual form of this equation is used which is a function of the discretization error of the primal problem and the truncation error of



(a) Uniform, 18016 cells (b) (τ^n), 18225 cells (c) (τ^s), 18059 cells



(d) (τ^s, D), 18137 cells (e) (τ^s, C), 18624 cells (f) $(\tau_a^s, \varepsilon_p)$, 17682 cells

Figure 6.3: Adapted meshes by different adaptation indicators for Poisson

the adjoint problem:

$$\left(Z - Z_h, L\left(U - U_h\right)\right)_D = \left(L\left(Z - Z_h\right), U - U_h\right)_D \tag{6.12}$$

which is obtained by integration by parts.

The second finest mesh for each refinement sequence is shown in Figure 6.3. As depicted in Figure 6.3a, there is unnecessary refinement on the uniformly refined mesh at regions not having remarkable contribution in the functional. However, the mesh resolution near the peak value in the primal solution is significantly higher than other parts for the adapted meshes. Using the discrete p-truncation error as the adaptation indicator, Figure 6.3b, results in irregular refinement far from the peak point; however, using the continuous p-truncation error provides a more reasonable mesh. The adapted mesh based on the continuous p-truncation error weighted by the continuous adjoint

solution has high resolution at the peak point.

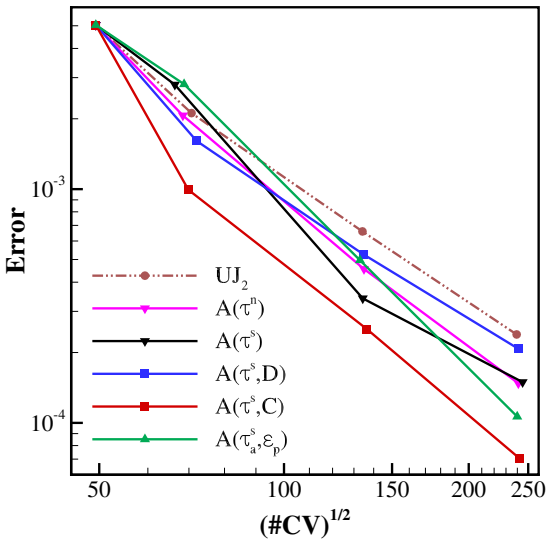


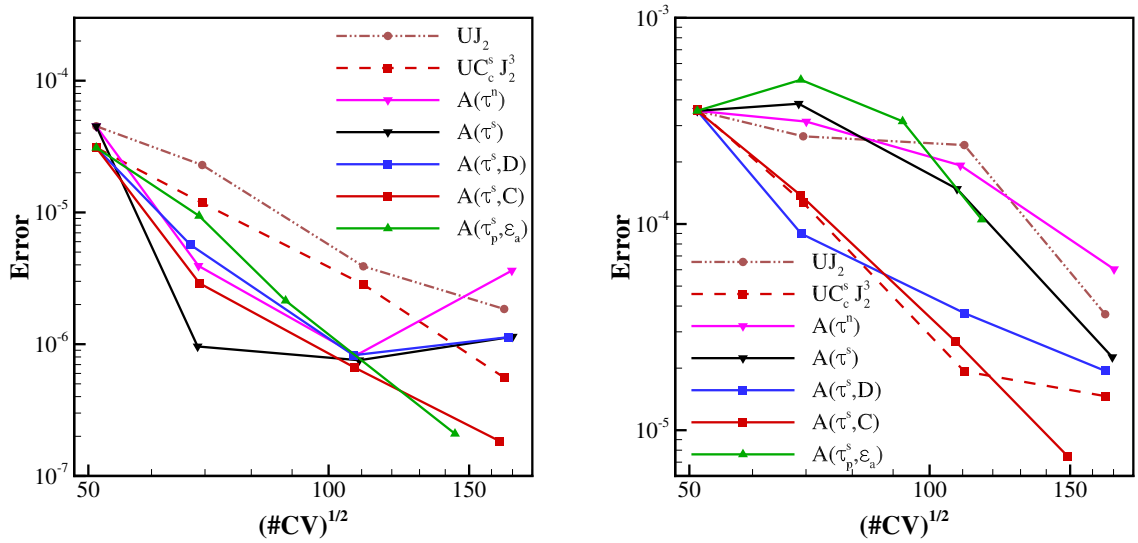
Figure 6.4: Discretization error of the primal problem on the adapted meshes for Poisson

Another point that needs to be considered is that the main goal of adaptation in this case is obtaining a more accurate functional value and the discretization error of the primal problem, for instance, does not behave in the same way as shown in Figure 6.2. The convergence of the discretization error of the primal problem with mesh refinement is shown in Figure 6.4. Although the discretization error for all adapted meshes are smaller than the second order solution on the uniformly refined mesh, the dash-dotted line, the convergence behaviors are not exactly the same as Figure 6.2. The discretization error on the adapted mesh based on the discretization error of the primal problem and the truncation error of the adjoint problem, the green line, which has the smallest error when output functional is calculated, is not the best one when discretization error is considered. This is not surprising since the goal of adaptation is improving the functional value and it does not necessarily guarantee that the discretization error or the error in calculating any other output functional is less on the same adapted mesh.

6.5 Subsonic Inviscid Flow on NACA 0012

Output functionals of drag and lift are considered for the subsonic inviscid flow around a NACA 0012 with $Ma_\infty = 0.5$ and angle of attack $\alpha = 2°$, the same test case as

discussed in Chapter 5. The exact drag and lift is considered as the second-order solution on a mesh with characteristic size thirteen times smaller in area than the finest uniformly refined mesh. The farfield boundary is a circle of radius 500 chords centered at the leading edge of the the airfoil and consequently, the errors arising from boundary placement is small. Theoretically, the exact drag value for this case should be zero; however, farfield effects keep the computed comparison value from being zero.



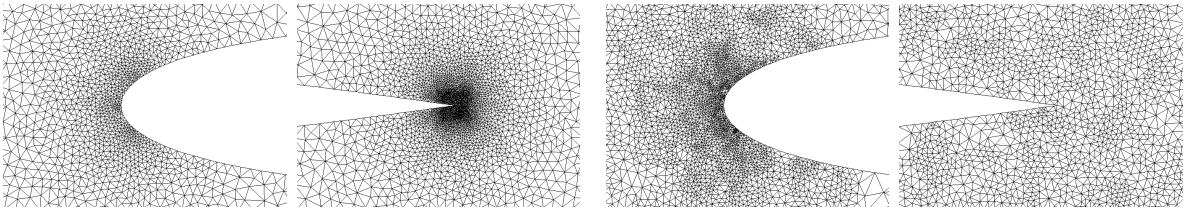
(a) Drag functional

(b) Lift functional

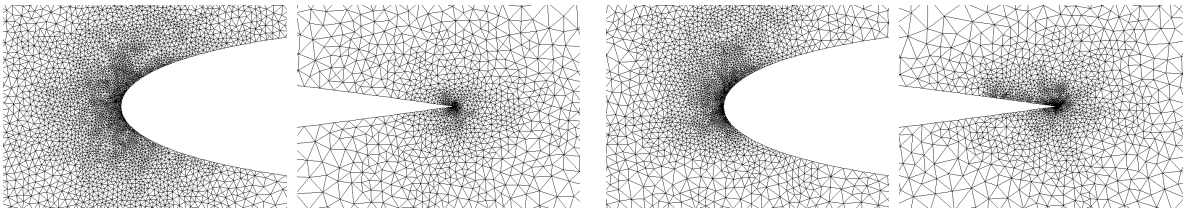
Figure 6.5: Convergence history for subsonic inviscid flow around NACA 0012 on the adapted meshes, $Ma_\infty = 0.5$, $\alpha = 2°$

The convergence results on the adapted meshes are shown in Figure 6.5 for both drag and lift functionals. The brown dash-dotted line is the non-corrected second-order functional on the uniformly refined mesh and the dashed red line is the corrected functional on the same mesh using the continuous p-truncation error estimate and the continuous adjoint solution as the best corrected functional. The solid lines are as discussed in Section 6.4 except that the adaptation indicator for the green line is the higher-order terms in Eq. 6.9. Using continuous p-truncation error estimate weighted by the continuous adjoint solution as the adaptation indicator, the solid red line, provides the best results for the lift functional and the second best for the drag functional. The fastest convergence rate for the drag functional is obtained using the dot product of the primal truncation error and the adjoint discretization error as the adaptation indicator; however, for the lift functional, this indicator is not as successful as for drag. Adjoint-

based mesh adaptation predicts more accurate results compared to residual-based mesh adaptation, as expected.

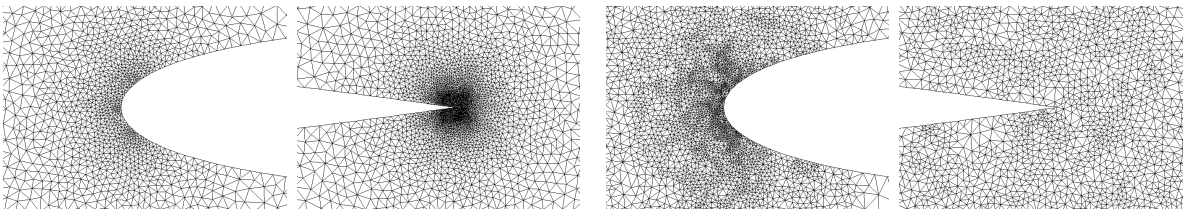


(a) Uniform, 27538 cells (b) (τ^s), 28925 cells

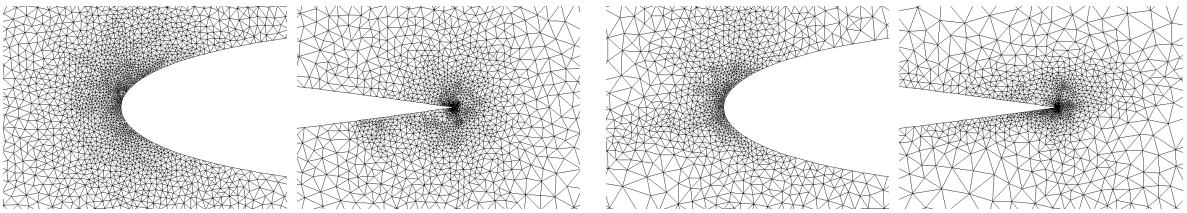


(c) (τ^s, C), 26808 cells (d) $\left(\tau_p^s, \varepsilon_a\right)$, 20734 cells

Figure 6.6: Adapted meshes by different adaptation indicators for subsonic inviscid flow around NACA 0012 for the drag functional, $Ma_\infty = 0.5$, $\alpha = 2^\circ$



(a) Uniform, 27538 cells (b) (τ^s), 28701 cells



(c) (τ^s, C), 22167 cells (d) $\left(\tau_p^s, \varepsilon_a\right)$, 13508 cells

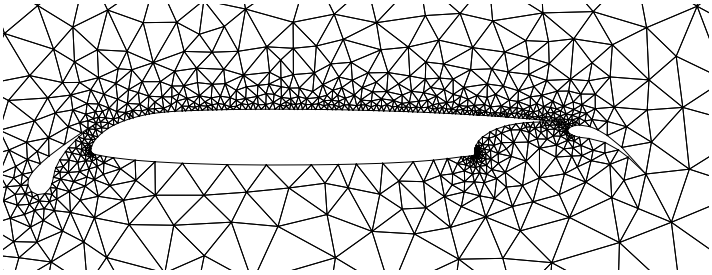
Figure 6.7: Adapted meshes by different adaptation indicators for subsonic inviscid flow around NACA 0012 for the lift functional, $Ma_\infty = 0.5$, $\alpha = 2^\circ$

The finest mesh for different adaptation indicators is shown in Figure 6.6 and 6.7 for drag and lift functionals, respectively. As both lift and drag are mainly affected by the mesh resolution at leading and trailing edges, the close-up view at these two regions

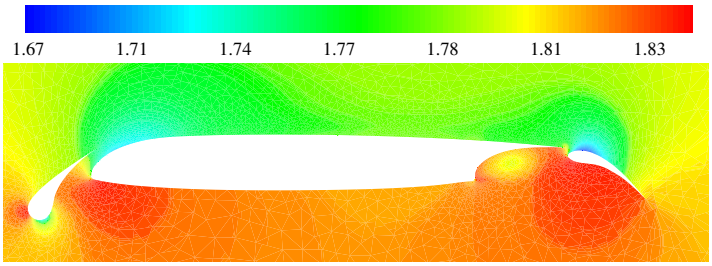
are shown here. The mesh is over-refined at a large region near the trailing edge only on the uniformly refined mesh; however, both trailing and leading edge are refined for adjoint-based and higher-order error-based mesh adaptation and the mesh resolution is finer in small regions near trailing edge. The residual-based mesh adaptation fails to have enough resolution at the trailing edge.

6.6 Subsonic Inviscid Flow on Multi-Element Airfoil

We now consider a more complicated geometry, a multi-element airfoil, with $Ma_\infty = 0.2$ and angle of attack $\alpha = 3°$. The second-order solution on a mesh with characteristic cell size eight times smaller in area than the finest mesh for which correction is applied is considered as the exact solution to the problem. The initial mesh consisting of 1877 cells is shown in Figure 6.8a. The pressure distribution for the primal problem is also depicted in Figure 6.8b.



(a) Close-up of multi-element airfoil mesh consisting of 1877 cells

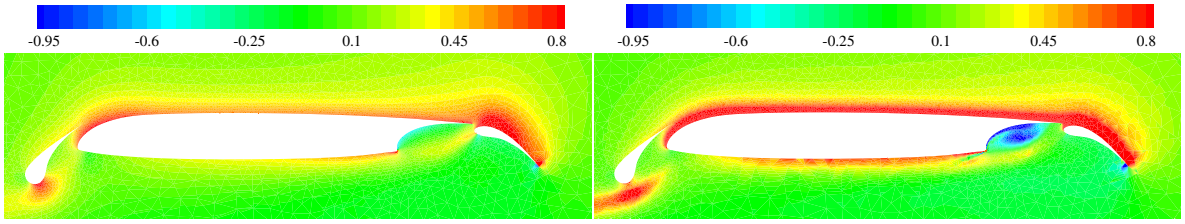


(b) Pressure distribution

Figure 6.8: Initial mesh and the primal solution distribution for subsonic inviscid flow on the multi-element airfoil, $Ma = 0.2$, $\alpha = 3°$

As this multi-element airfoil is a high-lift configuration design, the lift is consid-

ered as the output functional of interest. The adjoint energy component of the both continuous and discrete adjoint solutions for the lift functional are depicted in Figure 6.9. Similar to the comparison discussed in Chapter 5, the discrete and continuous adjoint solutions are qualitatively similar. The time comparison for these adjoint solution is shown in Figure 6.10 where the time for the second-order primal problem, the second-order continuous adjoint and the discrete adjoint problems used for correction and the CPU time required to obtain the C^1 interpolation of the solution for finding the continuous p-TE estimate are compared. The four meshes shown in this figure are the uniformly refined meshes. Time comparison shows that the post processing time required for C^1 interpolation of the solution is small compared to the solution time, and scales linearly with problem size. Sensitivity of the functional with respect to the primal solution is only a function of pressure in the case of inviscid flow. Therefore, the time required for solving the system of equations to obtain the discrete adjoint solution is significantly smaller than the time required to solve the continuous adjoint PDE for the inviscid case, because there is no physical dissipation to enhance the stability of the solution. For this inviscid case, the continuous adjoint solver oscillates before getting close to the solution, after which a larger time step can be used to obtain the converged solution.



(a) Continuous adjoint for lift functional (b) Discrete adjoint for lift functional

Figure 6.9: Normalized energy component of the adjoint solution for subsonic inviscid flow on the multi-element airfoil, $Ma = 0.2$, $\alpha = 3^\circ$

The convergence results for the lift functional are depicted in Figure 6.11a. Both the non-corrected and the corrected functionals using the continuous p-truncation error estimate and continuous adjoint solution on the uniformly refined mesh are compared to the adapted meshes. Not surprisingly, on the adjoint-based adapted meshes the functional is more accurate and converges faster to the exact value compared to the residual-based adapted meshes. Furthermore, the adapted meshes generated using the continuous p-truncation error are more accurate than the discrete p-truncation error.

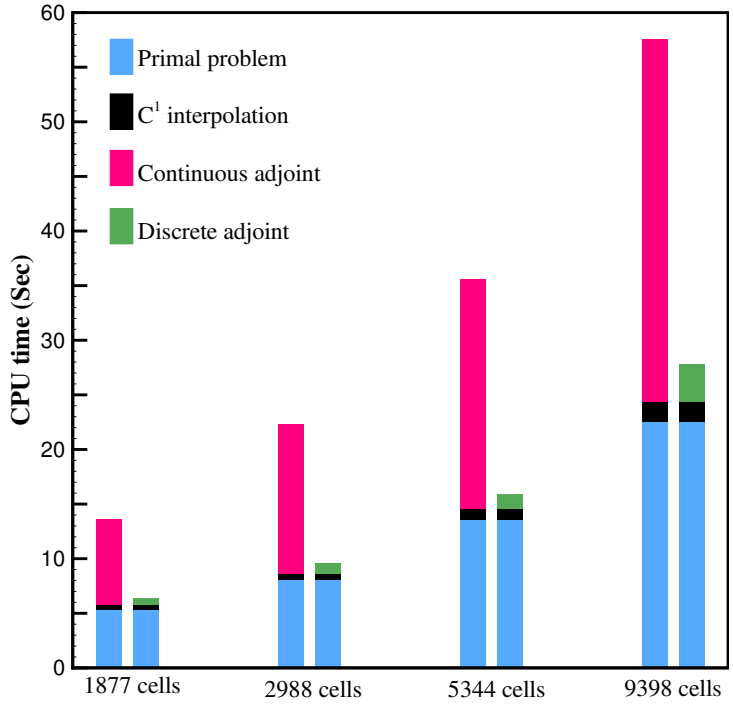
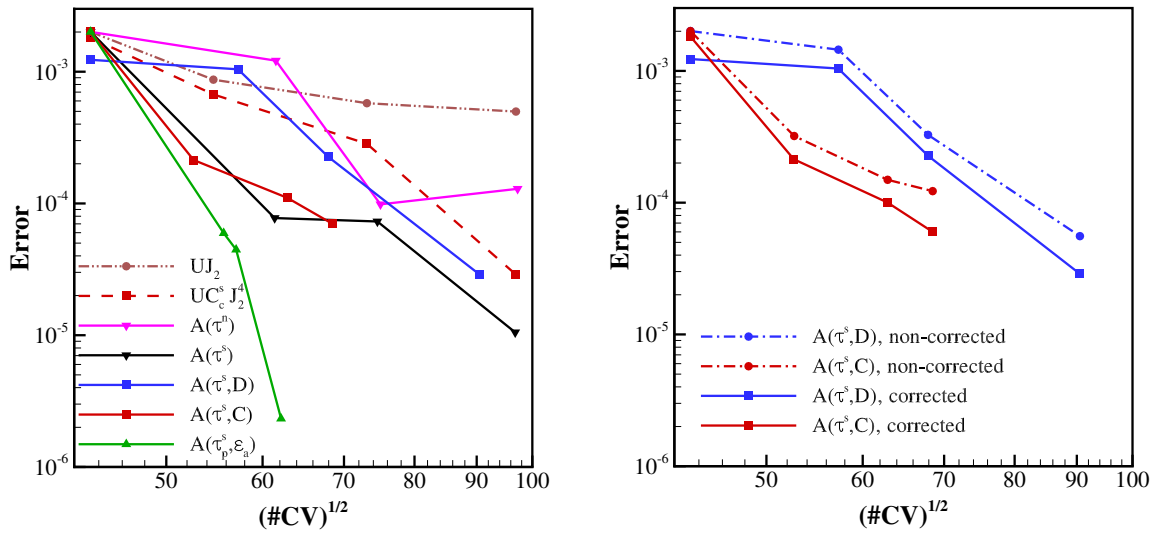


Figure 6.10: CPU time comparison for primal and adjoint solutions and C^1 interpolation of the solution for subsonic inviscid flow on the multi-element airfoil, $Ma = 0.2$, $\alpha = 3^\circ$

Using the higher-order term as the adaptation indicator provides the most accurate results with the fastest convergence rate.

The results for adjoint-based adapted meshes shown in Figure 6.11a are the corrected solutions and these corrected functionals are compared to the original non-corrected ones in Figure 6.11b for both the discrete and continuous adjoint solutions. Two points need to be considered here. First, since the same scale is used for both figures, comparing the second-order non-corrected functional on the uniformly refined mesh with non-corrected adjoint-based adapted mesh, the dash-dotted lines in Figure 6.11b, shows that the adjoint-based adapted mesh provides a considerably more accurate functional compared to the uniformly refined mesh even without correction. Second, correcting the functional calculated on the adapted mesh provides us even a more accurate value which converges to the exact value faster.

The finest mesh for each series is shown in Figure 6.12. Although on the uniformly refined mesh, all cells around the solid wall get refined, the adapted mesh focuses mainly on the leading and trailing edges of each element, the regions where the lift is affected significantly. This is more noticeable when adjoint-based mesh adaptation or



(a) Adaptation indicators

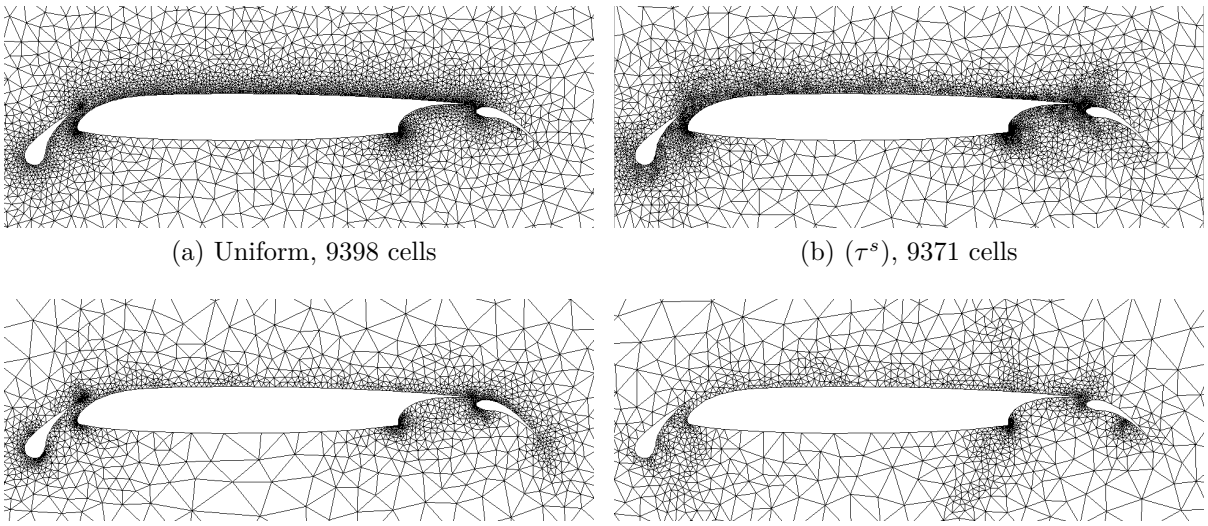
(b) Correction effect

Figure 6.11: Convergence history for subsonic inviscid flow around multi-element airfoil on the adapted meshes for the lift functional, $Ma_\infty = 0.2$, $\alpha = 3°$

adaptation based on higher-order error in the correction term is considered. That is the reason why a more accurate functional is obtained on the adapted meshes using these two adaptation indicators in spite of the fact that the total number of cells is less than the uniformly refined mesh.

6.7 Subsonic Viscous Flow on NACA 0012

Next, we consider viscous subsonic flow around a NACA 0012 airfoil with a free-stream Mach number of $Ma_\infty = 0.5$, a Reynold number of $Re = 5000$, and an angle of attack $\alpha = 1°$. The second-order solution on a mesh with characteristic cell size thirteen times smaller in area than the finest mesh for which correction is applied is considered as the exact solution to the problem. The farfield boundary is a circle of radius 100 chords centered at the leading edge of the the airfoil and consequently, the errors arising from boundary placement are small. As viscous flow is studied, the total drag is considered as the output functional. Discrete and continuous adjoint solution distributions for this case have been discussed in Chapter 5 and in this section, we only compare the time required to obtain the second-order primal and adjoint solutions to the time required to obtain the C^1 interpolation of the solution. Figure 6.13 shows that the time required



(a) Uniform, 9398 cells

(b) (τ^s), 9371 cells

(c) (τ^s, C), 4687 cells

(d) $\left(\tau_p^s, \varepsilon_a\right)$, 3861 cells

Figure 6.12: Adapted meshes by different adaptation indicators for subsonic inviscid flow around multi-element airfoil for the lift functional, $Ma_\infty = 0.2$, $\alpha = 3^\circ$

for obtaining the C^1 interpolation of the solution is much smaller the time for solving the primal problem. However, the time for solving the continuous and discrete adjoint solutions are comparable. As a consequence of the viscous terms, convergence of the continuous adjoint PDE is faster compared to the inviscid case and it does not oscillate before getting close to the solution. Furthermore, the sensitivity of the functional with respect to the solution is more complicated in the case of viscous flow and consequently solving the linear system to obtain the discrete adjoint solution needs more time.

The convergence results for the drag functional are depicted in Figure 6.14a. Similar to the other test cases being studied so far, all adapted meshes outperform the non-corrected functional on the uniformly refined mesh, the brown dash-dotted line, in terms of the error magnitude and convergence rate. Moreover, all of them except for the residual-based adaptation using the discrete p-truncation error, the magenta line, outperform the corrected functional on the uniformly refined mesh, the dashed red line. The results for the adjoint-based adapted meshes using the continuous adjoint solution are more accurate than residual-based ones and the adjoint-based using the discrete adjoint solution. The convergence behavior is quite similar to the adapted mesh based on the higher-order terms in the correction term, the green line.

The corrected functionals are compared to the non-corrected one and the result is shown in Figure 6.14b; the behavior is the same as discussed in Section 6.6.

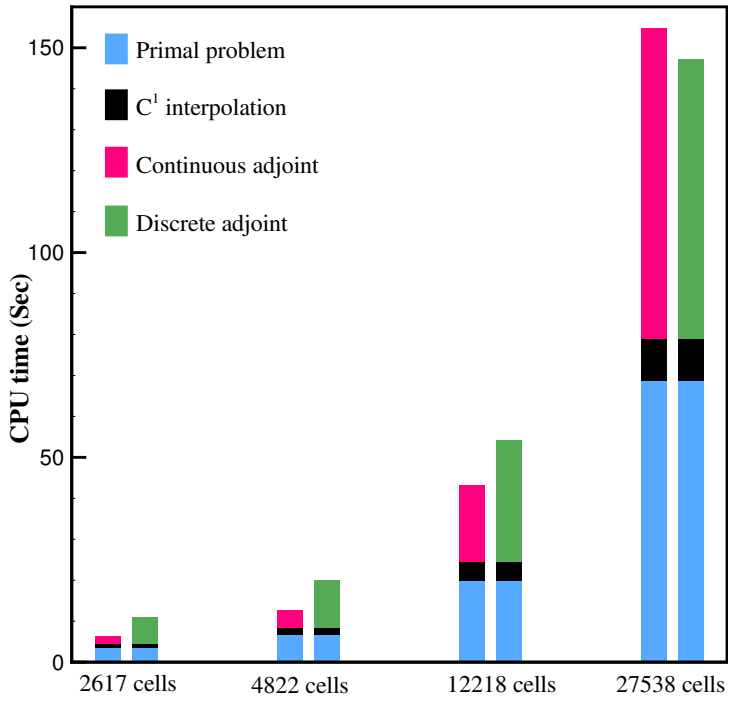
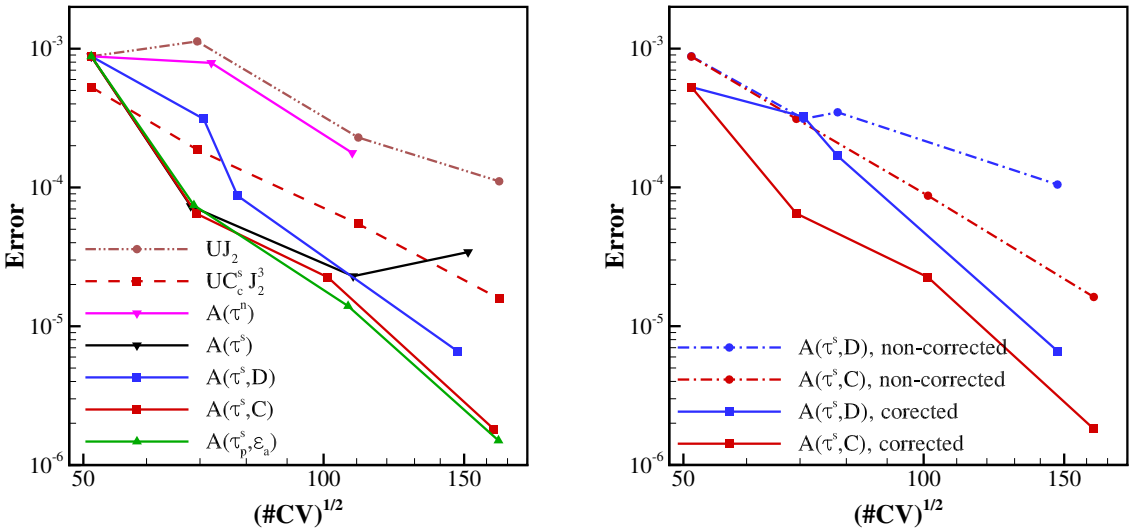


Figure 6.13: CPU time comparison for primal and adjoint solutions and C^1 interpolation of the solution for subsonic viscous flow around NACA 0012, $Ma_\infty = 0.5$, $\alpha = 1°$, $Re = 5000$

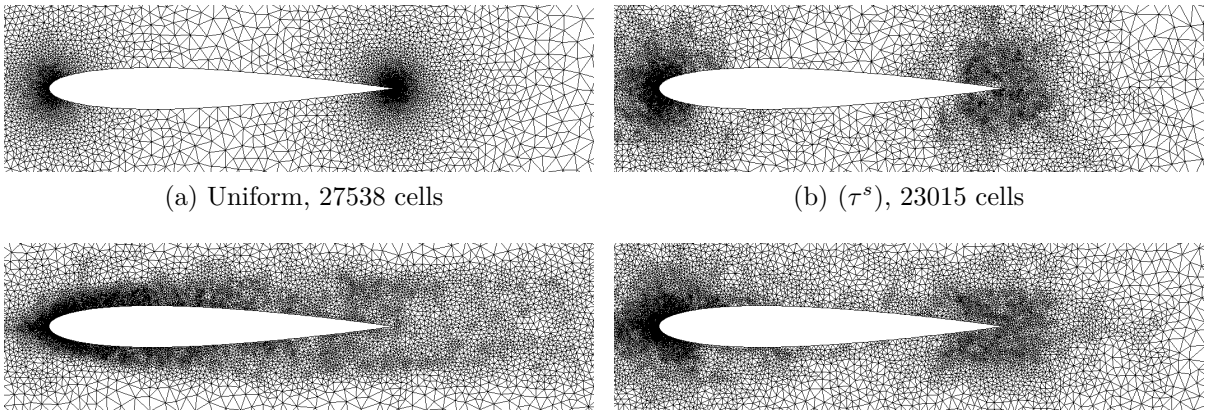


(a) Adaptation indicators

(b) Correction effect

Figure 6.14: Convergence history for subsonic viscous flow around NACA 0012 on the adapted meshes for the drag functional, $Ma_\infty = 0.5$, $\alpha = 1°$, $Re = 5000$

The finest mesh for each case is shown in Figure 6.15. Although the uniformly refined mesh focuses only on the leading and trailing edges, the adapted mesh has enough resolution along the boundary layer. The mesh resolution along the boundary layer is higher for the adjoint-based adapted mesh and the mesh adapted based on the higher-order error in the correction term. The wake region is another feature of the viscous flow and the adjoint-based adapted mesh is more successful in resolving this part of the flow.



(a) Uniform, 27538 cells

(b) (τ^s), 23015 cells

(c) (τ^s, C), 26686 cells

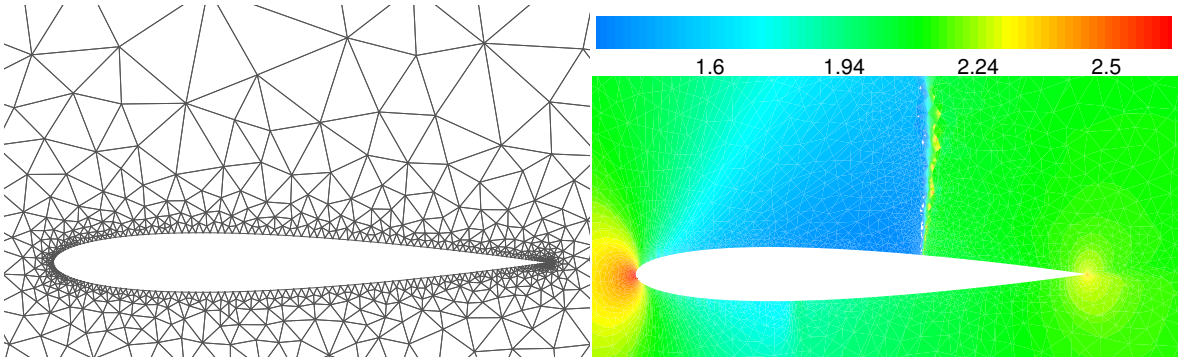
(d) $\left(\tau_p^s, \varepsilon_a\right)$, 27426 cells

Figure 6.15: Adapted meshes by different adaptation indicators for subsonic viscous flow around NACA 0012 for the drag functional, $Ma_\infty = 0.5$, $\alpha = 1^\circ$, $Re = 5000$

6.8 Transonic Inviscid Flow on NACA 0012

Finally, inviscid transonic flow around the NACA 0012 airfoil with a free-stream Mach number of $Ma_\infty = 0.8$ and an angle of attack $\alpha = 1.25^\circ$ is considered. The second-order solution on a mesh with characteristic cell size eight times smaller in area than the finest mesh for which correction is applied is considered as the exact solution to the problem. The farfield boundary is a circle of radius 100 chords centered at the leading edge of the airfoil. The initial mesh consisting of 1998 cells is shown in Figure 6.16a. The pressure distribution for the primal problem is also depicted in Figure 6.16b.

The discrete adjoint solution distribution for the lift functional is shown in Figure 6.17. For this case, due to some robustness issues for the continuous adjoint solution, we just show the results for the discrete adjoint.



(a) Close-up of NACA 0012 airfoil mesh consisting of 1998 cells

(b) Normalized energy distribution

Figure 6.16: Initial mesh and the primal solution distribution for transonic inviscid flow on NACA 0012 airfoil, $Ma = 0.8$, $\alpha = 1.25°$

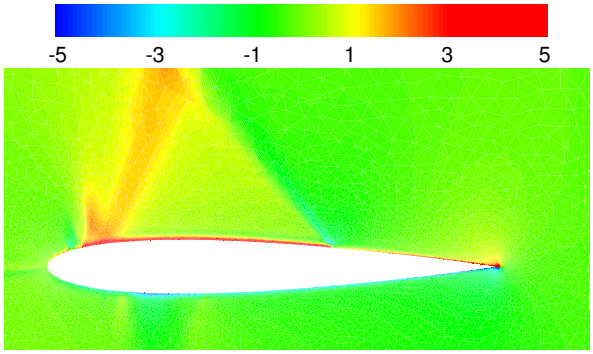


Figure 6.17: x-momentum adjoint solution for the lift functional, transonic inviscid flow on NACA 0012 airfoil, $Ma = 0.8$, $\alpha = 1.25°$

The convergence results for the adapted meshes are shown in Figure 6.18. The calculated functional on the adjoint-based adapted meshes and the residual-based adapted meshes using the continuous p-truncation error are more accurate compared to non-corrected and corrected functionals on the uniformly refined mesh. For this test case, the discrete adjoint-based mesh adaptation using the discrete and continuous p-truncation error are compared and the continuous p-truncation error estimate is again more accurate.

The finest mesh for each case is shown in Figure 6.19. The strong shock on the upper surface of the airfoil is resolved using all three adaptation indicators; however, using the continuous p-truncation error, either for residual-based or adjoint based adapted meshes, a sharper shock region is detected. The residual-based indicator using the

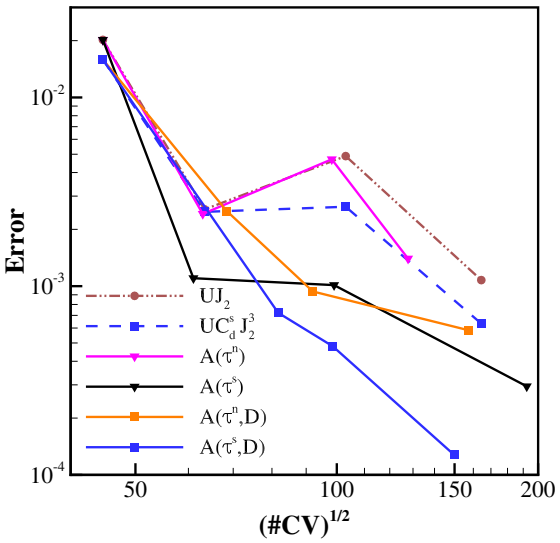
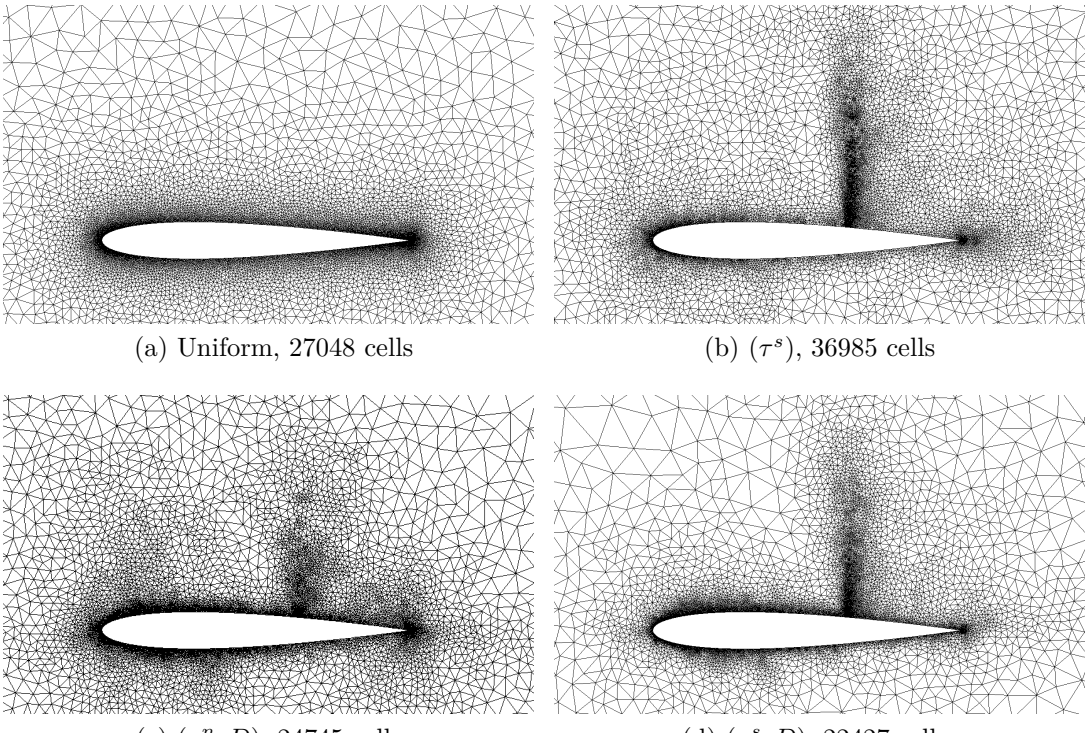


Figure 6.18: Convergence history for transonic inviscid flow around NACA 0012 on the adapted meshes for lift functional, $Ma_\infty = 0.8$, $\alpha = 1.25°$



(a) Uniform, 27048 cells

(b) (τ^s), 36985 cells

(c) (τ^n, D), 24745 cells

(d) (τ^s, D), 22427 cells

Figure 6.19: Adapted meshes using different adaptation indicators for transonic inviscid flow around the NACA 0012 for the lift functional, $Ma_\infty = 0.8$, $\alpha = 1.25°$

continuous p-truncation error focuses mainly on the upper shock but the adjoint-based indicator using the continuous p-truncation error refines the weak shock on the lower surface as well.

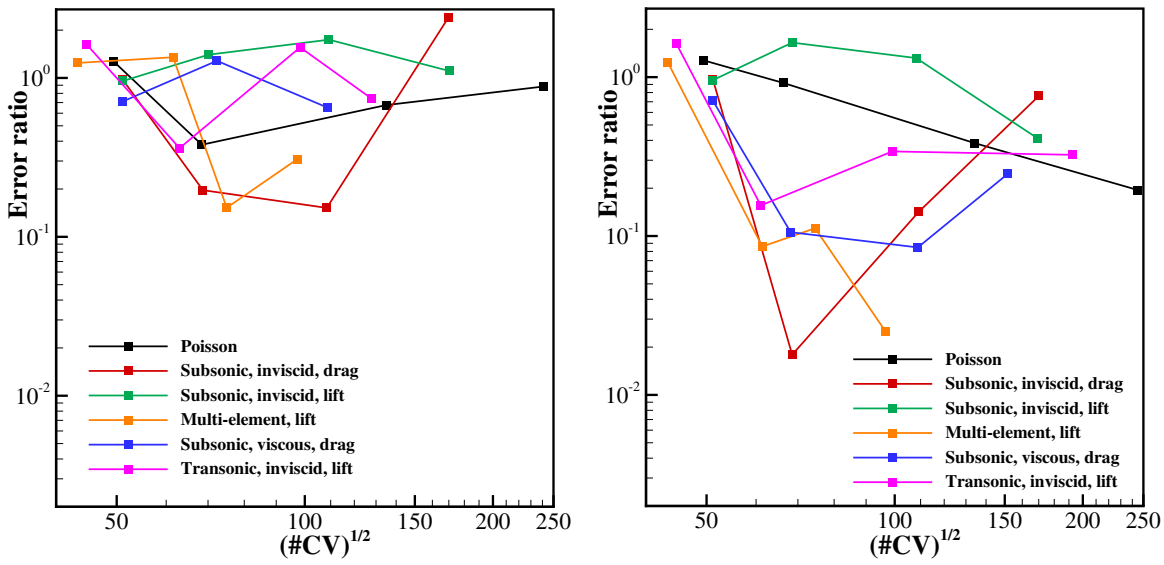
6.9 Discussion

A summary of the results for all test cases is as follow:

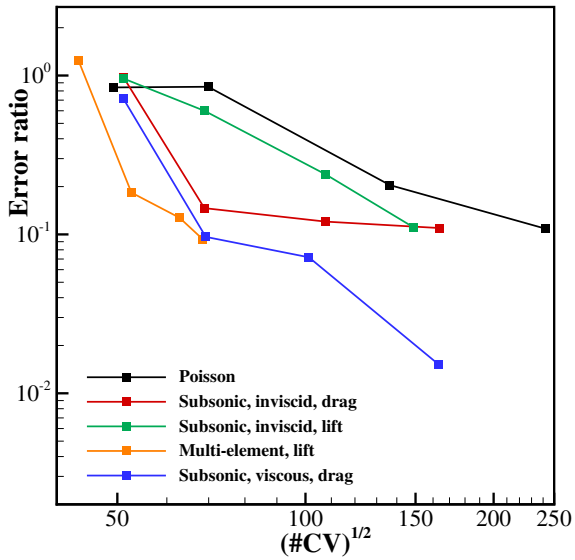
- For all test cases studied in this chapter, the functional on the adapted mesh is calculated more accurately compared to the non-corrected one calculated on the uniformly refined mesh.
- The results for residual-based adaptation using the continuous p-truncation error, adjoint based adaptation, and adaptation based on the higher-order error in the correction term are more accurate compared to the corrected functional on the uniformly refined mesh almost for all cases except for the lift functional for subsonic inviscid flow on the NACA 0012.
- Residual-based adaptation using the discrete p-truncation error does not necessarily provide a significantly more accurate functional than the corrected one on the uniformly refined mesh.
- Residual-based mesh adaptation using continuous p-truncation error is much better than using the discrete p-truncation error for all test cases.
- Adjoint-based mesh adaptation outperforms the residual-based mesh adaptation for all test cases.
- Adjoint-based mesh adaptation using the continuous adjoint solution is always better than the discrete adjoint, or at least, they are comparable.
- The results using the higher-order error in the correction term as the adaptation indicator are more accurate or comparable to the adjoint-based mesh adaptation using the continuous adjoint solution for almost all test cases except for the lift functional for subsonic inviscid flow on the NACA 0012.

In summary, mesh adaptation based on the continuous p-truncation error weighted by the continuous adjoint solution, for cases where the adjoint solution is easy to calculate,

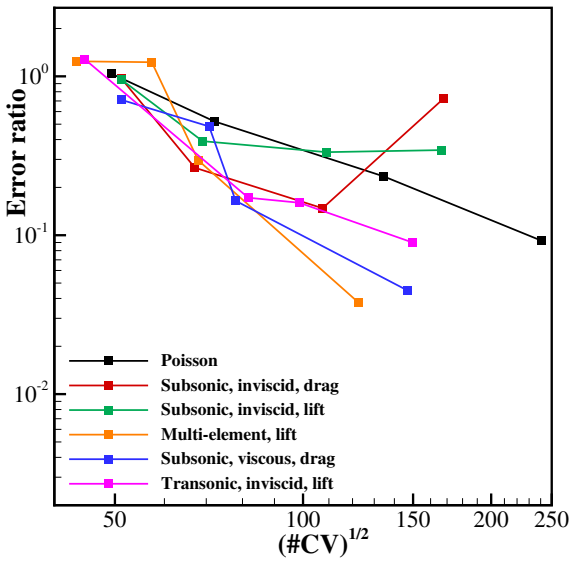
provides the best results. When this is feasible, using the higher order error in the correction term is a very good choice as well. Otherwise, if the continuous adjoint solution is not available, or if it is expensive to compute, using the continuous p-truncation error weighted by the discrete adjoint solution is also helpful. The same comparison as described at the end of Chapter 6 is shown here for the adapted meshes. The error in calculating the second-order non-corrected functional on the uniformly refined mesh is considered as the reference value and the ratio of the error of the functional on the adapted meshes to this reference value is shown here. In Chapter 5, the same meshes were used, but in the case of the adapted meshes, the best line that fits the four points is considered and the reference value corresponding to the mesh with the same number of control volumes, or the non-corrected functional error, is found on that line. The results are summarized in Figure 6.20. Since the same scale is used for all five plots, comparing the residual based adaptation to the adjoint-based adaptation shows that the magnitude of the error for the adapted meshes using the truncation error weighted by the adjoint solution is smaller. Moreover, adaptation based on continuous adjoint solution provides more consistent results for all test cases. Adaptation based on the higher-order error in the correction terms provides the functional with the smallest error and the results are quite consistent for all test cases except for the lift functional for the subsonic inviscid flow around NACA 0012.



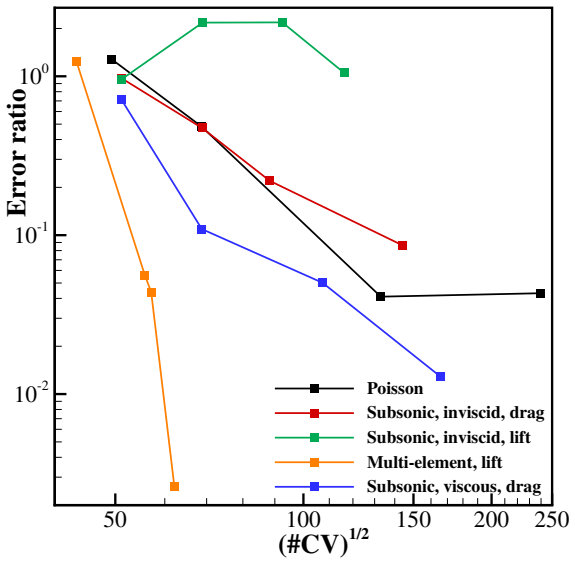
(a) (τ^n) (b) (τ^s)



(c) (τ^s, C)



(d) (τ^s, D)



(e) $\left(\tau_p^s, \varepsilon_a\right)$

Figure 6.20: The error ratio for different adaptation indicators for all test cases based on second-order functional

Chapter 7

Concluding Remarks

7.1 Summary

As the capabilities of CFD software grow, the size and complexity of industrial CFD problems are increasing as well. These new large scale problems are challenging. As part of analyzing these flows, important flow features should be identified and resolved and the accuracy of the solution should be characterized. CFD software users would benefit from automatic numerical error control through defect correction, *a posteriori* estimation of the error in outputs of engineering interest, and goal-based adaptation. Taken together, these improvements would pave the way for truly mature CFD, in which users would need to specify only geometry, flow conditions, output quantities of interest, and an error tolerance.

Accurate estimation of the truncation error plays an important role in error quantification and consequently, developing a reliable truncation error estimator for unstructured meshes, which are used to handle the complex problem geometry, is needed. Truncation error can be used to improve the flow solution through defect correction, output functional error estimation and correction, and goal-based adaptation.

There are multiple options for estimating the truncation error. If the exact solution is known, we can calculate the flux integrals based on the exact solution to get the exact truncation error. However, this is not possible for general problems with unknown exact solution. Instead, a higher-order flux integral can be calculated based on the lower-order solution. This method, called the discrete p-TE method, includes the leading term in the truncation error and consequently can be considered as an estimate of the truncation error.

We have demonstrated that — in contrast with the structured mesh case for which the truncation error has the same asymptotic order of accuracy as the discretization error — the truncation error for unstructured finite volume schemes is asymptotically larger than the discretization error, which in turn is typically of the same order as the solution approximation error in the scheme. Furthermore, the truncation error for

unstructured meshes is rough as a result of the irregular topology of the mesh.

The differences in asymptotic behavior of the truncation and discretization error for unstructured meshes and the features of the truncation error can be predicted by the eigendecomposition of the discrete problem. Our results based on the eigenanalysis of the truncation error have shown that the rough error modes dominate the unstructured mesh truncation error. The dominance of rough modes makes truncation error estimation more challenging when arbitrary unstructured meshes are used. If a perfect symmetric mesh is used, the truncation error is much smoother and easier to estimate accurately even in the high frequency modes. Nevertheless, it is not possible to generate perfectly symmetric meshes for real application cases. Therefore, the dominance of rough error modes in the unstructured mesh truncation error suggests the need to develop a smooth truncation error estimate that can be computed using quantities already available in a typical finite volume flow solver. As mentioned, the discontinuous jumps of the coefficients of the leading terms in the Taylor series expansion of the error from one control volume to another is responsible for the rough distribution of the truncation error. Since we are using the $p-$truncation error approach for estimating the truncation error, which computes the higher-order flux integration based on the lower-order solution, two different values are obtained from reconstruction of the solution at the quadrature points at the common face of the two neighboring cells where the flux integral is calculated. By using a continuous interpolation of the solution, we can obtain a single value for the flux vector from two neighboring cells. The smooth solution and consequently the smooth flux vector gives us an estimation of the truncation error based on the continuous solution which was shown to provide us with p-sharp convergence.

Two different approaches were used to obtain a continuous interpolation of the solution: using CGM, a 3D CAD query engine for meshing, and C^1 interpolation of the solution. Using CGM, the solution at each point is considered as the third component and consequently, a smooth 3D surface with the coordinates of the 2D mesh and the solution is obtained in which there is a single solution at the quadrature points at the common face of the two adjacent cells. Although this method gives us a unique solution, this is not sufficient, as the gradient is not calculated accurately and we need a unique gradient at the quadrature points as well. Consequently, we have presented a C^1 interpolation scheme to obtain continuous solution and gradients at the quadrature points.

The performance of the original discrete p-TE method is compared to the perfor-

mance of the continuous p-truncation error for defect correction, output functional correction and mesh adaptation for a variety of test cases, including model problems with a manufactured solution, Euler and Navier-Stoke equation with a manufactured solution, and realistic flow test cases.

7.2 Conclusions

The idea of defect correction is that if the exact truncation error is added to the source term of the problem and the new discrete equation with the modified source term is solved, the exact solution is obtained. Alternatively, if the p-TE estimate obtained based on higher-order flux integral, which is an estimation of the truncation error, is added to the source term of the problem, the defect corrected solution should ideally be p-sharp, implying that it converges as fast as the higher-order reconstructed solution used in calculating the flux integral. This is the result for structured meshes. For unstructured meshes, our results show that if the original p-TE method is used for defect correction, either for the Poisson equation with diffusive fluxes or the advection equation with convective fluxes, p-sharp convergence is not obtained. Moreover, the magnitude of the discretization error of the corrected solution is more than the uncorrected one. Consequently, the continuous p-TE estimate of the truncation error is used for defect correction. Using CGM to interpolate the solution is not able to improve the performance of defect correction significantly either. For both diffusive and convective fluxes, using C^1 interpolation to compute the continuous p-truncation error estimate provides a corrected solution with smaller discretization error, but the convergence rate is still not as good as expected.

As a consequence of the poor performance of CGM for defect correction, we continued with using C^1 interpolation for functional correction and goal-based mesh adaptation. Although C^1 interpolation of the solution is not able to improve the performance of defect correction significantly, particularly in terms of convergence rate, this does not necessarily mean that it is not able to improve an output functional or effectively adapt the mesh.

The truncation error, when multiplied by the adjoint solution, is used to compute a correction term for an output functional of interest. Both discrete and continuous adjoint solutions can be used. The discrete adjoint is obtained by solving a linear system where the left-hand side is the transpose of the Jacobian matrix and the right-

hand side is the sensitivity of the output functional with respect to the primal solution. The continuous adjoint solution, on the other hand, is an independent PDE which needs to be discretized and solved separately. The performance of output functional correction using both discrete and continuous C^1 interpolated truncation error weighted by the discrete and continuous adjoint solutions were compared. Scalar equations (the Poisson and advection equations) with a manufactured solution and systems of equations (the Euler and Navier-Stokes equations) using both manufactured solutions and real flow test cases are considered. Our results showed that using the discrete truncation error weighted by either continuous or discrete adjoint solution is not helpful in improving the output functional in terms of convergence rate. On the other hand, using the C^1 interpolated solution improves the functional significantly. The second-order solution is used to find a continuous flux integral based on the third or fourth-order reconstruction, giving third or fourth-order convergence for the functional while only solving the primal and adjoint problems at second-order. Using fourth-order flux integration based on the third-order solution provides us with similar improvement in convergence rate and for both cases, the magnitude of the functional error is also less than the magnitude of the uncorrected functional error. More accurate corrected functionals are obtained when correction is based on the continuous adjoint since this gives a more accurate adjoint solution compared to solving a linear system which gives the discrete adjoint solution. As a result, more consistent and accurate results for a range of cases are obtained using the continuous adjoint. Computational time for the continuous adjoint problem is about the same as computational time for the primal problem. For the discrete adjoint problem, the computational time is the time of solving a linear system, so about the cost of one time step for the primal. Sensitivity of the functional with respect to the primal solution is only a function of pressure in the case of inviscid flow. Therefore, the time required for solving the system of equations to obtain the discrete adjoint solution is significantly smaller than the time required to solve the continuous adjoint PDE for the inviscid case where there is not any viscous term which enhances the stability of the problem. For the inviscid case, the residual for the continuous adjoint solver oscillates before getting close to the solution, after which a larger time step can be used to obtain the converged solution. However, for the viscous case, the time for solving the continuous and discrete adjoint solutions are comparable. As a consequence of the viscous terms, convergence of the continuous adjoint PDE is faster compared to the inviscid case. Furthermore, the sensitivity of the functional with respect to the solution is more complicated in the case of viscous flow and consequently finding the discrete

adjoint solution requires more time.

This significant improvement in functional correction is obtained just by C^1 interpolating the primal solution and finding the truncation error based on this interpolated solution and then multiplying the truncation error by the adjoint solution of the same order as the primal solution. It is not necessary to find the C^1 interpolation of the adjoint solution to obtain significant improvement in convergence rates. By estimating the truncation error based on a C^1 interpolated solution, higher order convergence of the functional is obtained by using the lower order solution. Hence, paying the extra cost of computing the higher-order solution to obtain a more accurate functional which converges to the exact value with higher-order rate is not required and the higher-order convergence is obtained. Time comparison showed that the post processing time required for C^1 interpolation of the solution is small compared to the solution time, and scales linearly with problem size. As a result, it is a reasonable post-processing step to obtain the higher-order convergence.

The truncation error can also be used as an indicator for mesh adaptation, called residual-based adaptation, or can be weighted by the adjoint solution, called adjoint-based mesh adaptation. The adaptation indicator for the adjoint-based adaptation is the same as the correction term used for functional correction. Furthermore, the higher-order error in the correction term can be used as an adaptation indicator. The effectiveness of different adaptation indicators using discrete or continuous truncation error were compared. Different test cases were considered. For all test cases studied, the functional on the adapted mesh is calculated more accurately compared to the non-corrected one calculated on the uniformly refined mesh. This result holds for residual-based adaptation using the continuous p-truncation error, adjoint based adaptation, and adaptation based on the higher-order error in the correction term. Although residual-based adaptation using the original discrete p-truncation error estimate does not necessarily provide a significantly more accurate functional than the corrected one on the uniformly refined mesh, residual-based mesh adaptation using the C^1 interpolated continuous truncation error always provide a more accurate functional compared to the functional calculated on the uniformly refined mesh. Our results show that adjoint-based mesh adaptation out performs the residual-based mesh adaptation and using the continuous adjoint solution is always better than the discrete adjoint, or at least, they are comparable. The results using the higher-order error in the correction term as the adaptation indicator are more accurate or comparable to the adjoint-based mesh adaptation using the continuous adjoint solution.

The following points summarize the main contributions of this thesis:

- We have developed a reliable truncation error estimate for unstructured meshes using quantities already available in a typical finite volume flow solver. This estimate of the truncation error should be general and not dependent on the geometry, the properties of the solution, or the governing equation such that it can be easily generalized to any problem. The method of our choice is the p-TE method.

- As the original discrete p-TE method is not able to provide p-sharp convergence for output functional correction or mesh adaptation, we developed a C^1 interpolation of the solution which gives us a unique solution and gradients at the quadrature point at the common face of the two neighboring cells where the flux function is evaluated.

- Evaluating the C^1 interpolation of the solution is a cheap post-processing step which uses the data already available for the lower-order solution and using a set of basis functions which needs to be solved only once and can be used for any problem regardless of the geometry, the governing equations or other properties of the solution.

- The truncation error estimate calculated using C^1 interpolation of the solution can be multiplied by the adjoint solution to get the correction term used in output functional correction. Both the discrete and continuous adjoint solutions can be used. Using the continuous adjoint solution for correcting the output functional provides more consistent results and more accurate functionals compared to the discrete adjoint solution. For finding the discrete adjoint solution, we only need the sensitivity of the output functional with respect to the primal solution and the Jacobian matrix; however, finding the continuous adjoint solution requires discretizating the continuous adjoint PDE and setting the corresponding the boundary conditions. Once, the continuous adjoint PDE has been discretized and the proper boundary conditions are implemented, using the continuous adjoint solution is more recommended as it provides more consistent results.

- Using the truncation error estimate calculated using C^1 interpolation of the solution multiplied by the adjoint solution to correct the functional provides the p-sharp functional convergence which convergences to the exact value as fast as

the higher-order reconstruction order used in calculating the truncation error. On the other hand, using the original p-TE method using the discrete solution, the convergence of the functional is the same as the primal solution order. The higher-order convergence of the output functional is obtained only by solving both the primal and adjoint problems at lower-order, in contrast to the finite element method where the higher-order adjoint solution is required. Furthermore, it is not necessary to find the C^1 interpolation of the adjoint solution to get p-sharp convergence.

- The truncation error estimate calculated using C^1 interpolation of the solution can also be used as the adaptation indicator which provides faster convergence of the output functional compared to using the original estimate of the truncation error based on the discrete solution as the adaptation indicator.

- Adjoint-based mesh adaptation using the truncation error estimate calculated using C^1 interpolation of the solution weighted by the continuous adjoint solution, the same as the correction term, gives us more accurate functional which converges to the exact value faster. Even more accurate results are obtained by correcting the functional. Note that, the correction step does not require any further information and is available when adjoint-based adaptation is used.

- Significant improvement is obtained if adaptation is based on the higher-order error in the correction term. Again, no further calculation is required to obtain this higher-order error term.

7.3 Recommendations for Future Work

We examined both scalar and systems of equations for both manufactured solutions and real flow test cases. The real flow test cases involve inviscid and viscous flows, in the subsonic, transonic and supersonic flow regimes for simple and complex geometries. This work may be extended to 2D turbulent flows for which an anisotropic mesh is required to resolve the anisotropic features of the flow along the boundary layer near the solid wall. Using the same Argyris element to obtain the C^1 interpolation of the solution is easy; however, anisotropic curved meshes are required along the boundary layer to resolve the curved solid wall. In this case, a linear transformation form the physical

element to the reference element is not sufficient and a cubic coordinate transformation is needed.

For the transonic case, we only used the discrete adjoint for adjoint-based mesh adaptation. The continuous adjoint solution does not converge efficiently. We used the Lax-Friedrich method for calculating the continuous adjoint fluxes. Using a more accurate flux function may be useful in improving the robustness of the continuous adjoint for transonic flow.

The output functional correction and mesh adaptation using the continuous adjoint solution was more helpful compared to the discrete adjoint solution. If the discretization scheme is adjoint-consistent, the effectiveness of output correction and mesh adaptation are the same using the discrete and continuous adjoint solutions. Therefore, improving our discretization scheme such that it is adjoint consistent would also be helpful.

The interpolation scheme we developed can be extended to any other two-dimensional problem without any difficulties as it worked properly for both viscous and inviscid flows which involve diffusive and convective terms. Extending that to 3D problems involves finding the proper polynomials for the 3D reference element, similar to the Argyris element for 2D, and then solving the system of equations arising from the basis functions once to obtain the constants. The 3D reference element which provides C^1 continuity is described by Walkington [96].

Bibliography

[1] First International Workshop on High-Order CFD Methods. http://people.ku.edu/z651w035/hiocfd.html, 2012. Sponsored by the AIAA Fluid Dynamics Technical Committee, the US Air Force Office of Scientific Research, and the Deutsches Zentrum fur Luft- und Raumfahrt.

[2] S. Adjerid, K. D. Devine, J. E. Flaherty, and L. Krivodonova. A posteriori error estimation for discontinuous galerkin solutions of hyperbolic problems. *Computer methods in applied mechanics and engineering*, 191(11):1097–1112, 2002.

[3] M. Ainsworth and J.T. Oden. *A Posteriori Error Estimation in Finite Element Analysis*. Wiley-Interscience, John Wiley and Sons, 2000.

[4] D. Ait-Ali-Yahia, G. Baruzzi, W. G. Habashi, M. Fortin, J. Dompierre, and M. G. Vallet. Anisotropic mesh adaptation: Towards user-independent, mesh-independent and solver-independent CFD. Part II: Structured grids. *International Journal for Numerical Methods in Fluids*, 39(8):657–673, July 2002.

[5] F. Alauzet and O. Pironneau. Continuous and discrete adjoints to the Euler equations for fluids. *International Journal for Numerical Methods in Fluids*, 70:135–157, 2012.

[6] Dale A. Anderson, John C. Tannehill, and Richard H. Pletcher. *Computational Fluid Mechanics and Heat Transfer*. Series in Computational Methods in Mechanics and Thermal Sciences. Hemisphere Publishing Corporation, New York, 1984.

[7] J.H. Argyris, I. Fried, and D.W. Scharpf. The tuba family of plate elements for the matrix displacement method. *The Aeronautical Journal of the Royal Aeronautical Society*, 72:701–709, 1968.

[8] R. Balasubramanian and J.C. Newman III. Comparison of adjoint-based and feature-based grid adaptation for functional outputs. *International Journal for Numerical Methods in Fluids*, 53:1541–1569, 2007.

[9] Timothy J. Barth and Paul O. Frederickson. Higher order solution of the Euler equations on unstructured grids using quadratic reconstruction. AIAA paper 90-0013, January 1990.

[10] R. Becker and R. Rannacher. Weighted a posteriori error control in finite element methods. In *Proceedings of ENUMATH-97, Heidelberg*, 1998.

[11] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. *Acta Numerica*, 2001:1–102, 2000.

[12] K. Bohmer, P. Hemker, and H. Stetter. The defect correction approach, in K. Bohmer, H. Stetter (Eds), Defect Correction Methods: Theory and Application. *Springer, Berlin*, 1984.

[13] Gustavo C. Buscaglia and Enzo A. Dari. Anisotropic mesh optimization and its application in adaptivity. *International Journal for Numerical Methods in Engineering*, 40:4119–4136, 1997.

[14] Andrew W. Cary, Andrew J. Dorgan, and Mori Mani. Towards accurate flow predictions using unstructured meshes. In *Proceedings of the Nineteenth AIAA Computational Fluid Dynamics Conference*, 2009. AIAA paper 2009-3650.

[15] Phillippe G. Ciarlet. *The finite element method for elliptic problems*. Society for Industrial and Applied Mathematics, 2002.

[16] J. M. Derlaga, T. Phillips, C. J. Roy, and J. Borggaard. Adjoint and truncation error based adaptation for finite volume schemes with error estimates. In *In 53rd AIAA Aerospace Sciences Meeting, AIAA SciTech. American Institute of Aeronautics and Astronautics*, 2015.

[17] B. Diskin and J.L. Thomas. Accuracy analysis for mixed-element finite-volume discretization schemes. Technical Report 2007-08, National Institute of Aerospace, 2007.

[18] Boris Diskin and James Thomas. Notes on accuracy of finite-volume discretization schemes on irregular grids. *Applied Numerical Mathematics*, 60:224–226, 2010. Rebuttal of Svard 2008.

[19] Boris Diskin, James L. Thomas, Eric J. Nielsen, Hiroaki Nishikawa, and Jeffrey A. White. Comparison of node-centered and cell-centered unstructured finite-volume discretizations. Part I: Viscous fluxes. In *Proceedings of the Forty-Seventh AIAA Aerospace Sciences Meeting*, 2009. AIAA paper 2009-597.

[20] Boris Diskin, James L. Thomas, Eric J. Nielsen, Hiroaki Nishikawa, and Jeffrey A. White. Comparison of node-centered and cell-centered unstructured finite-volume discretizations: Viscous fluxes. *AIAA Journal*, 48(7):1326–1338, July 2010.

[21] J. Dompierre, M. G. Vallet, Y. Bourgault, M. Fortin, and W. G. Habashi. Anisotropic mesh adaptation: Towards user-independent, mesh-independent and solver-independent CFD. Part III. Unstructured meshes. *International Journal for Numerical Methods in Fluids*, 39(8):675–702, July 2002.

[22] R.P. Dwight. Heuristic a posteriori estimation of error due to dissipation in finite volume schemes and application to mesh adaptation. *Journal of Computational Physics*, 227:2845–2863, 2008.

[23] Krzysztof Fidkowski and David Darmofal. Reviw of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics. *AIAA Journal*, 49(4):673–694, 2011.

[24] L. Fox. Some improvements in the use of relaxation methods for the ordinary and partial differential equations. *Proc. Roy. Soc. London A*, 190:31–59, 1947.

[25] F. Fraysse, E. Valero, and J. Ponsin. Comparison of mesh adaptation using the adjoint methodology and truncation error estimates. *AIAA Journal*, 50(9):1920–1932, 2012.

[26] M. B. Giles and N. A. Pierce. An introduction to the adjoint approach to design. *Flow, Turbulence, and Combustion*, 65(3–4):393–415, 2000.

[27] Serge Gosselin. *Delaunay refinement mesh generation of curve-bounded domains*. PhD thesis, The University of British Columbia, Department of Mechanical Engineering, 2009.

[28] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, and M. G. Vallet. Anisotropic mesh adaptation: Towards user-independent, mesh-independent and solver-independent CFD. Part I: General principles. *International Journal for Numerical Methods in Fluids*, 32(6):725–744, March 2000.

[29] W. Hackbusch. Local defect correction method and domain docomposition techniques, in: K. Bohmer, H. Stetter, Defect Correction Methods: Theory and Applications. *Springer, Berlin*, 1984.

[30] R. Hartmann, J. Held, and T. Leicht. Adjoint-based error estimation and adaptive mesh refinement for the rans and k-w turbulence model equations. *Journal of Computational Physics*, 230(11):4268–4284, 2011.

[31] Ralf Hartmann. Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier–Stokes equations. *International Journal for Numerical Methods in Fluids*, 51(9–10):1131–1156, 2006.

[32] Ralf Hartmann and Paul Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. *Journal of Computational Physics*, 183(2):508–532, 2002.

[33] Ralf Hartmann and Paul Houston. Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws. *SIAM Journal of Scientific Computing*, 24(3):979–1004, 2003.

[34] Marcelo Hayashi, Marco Ceze, and Ernani Volpe. Characteristic-based boundary conditions for the Euler adjoint problem. *International Journal for Numerical Methods in Fluids*, 71:1297–1321, 2013.

[35] Alireza Jalali. Truncation error analysis of unstructured finite volume discretization schemes. Master's thesis, The University of British Columbia, Department of Mechanical Engineering, 2012.

[36] Alireza Jalali and Carl Ollivier-Gooch. Accuracy assessment of finite volume discretizations of diffusive fluxes on unstructured meshes. In *Proceedings of the Fiftieth AIAA Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics, 2012. AIAA Paper 2012-0608.

[37] Alireza Jalali, Mahkame Sharbatdar, and Carl Ollivier-Gooch. Accuracy assessment of finite volume discretization schemes on unstructured meshes. *Computers and Fluidss*, 101:220–232, 2014.

[38] Anthony Jameson, Luigi Martinelli, and Niles A. Pierce. Optimum aerodynamic design using the Navier-Stokes equations. *Theoretical and Computational Fluid Dynamics*, 10(1–4):213–237, 1998.

[39] Antony Jameson. Aerodynamic design via control theory. *Journal of Scientific Computing*, 3(3):233–260, 1988.

[40] Antony Jameson, Sriram Shankaran, and Luigi Martinelli. Continuous Adjoint Methods for Unstructured Grids. *AIAA Journal*, 46:1226–1239, 2008.

[41] O. A. Karakashian and F. Pascal. A posteriori error estimates for a discontinuous galerkin approximation of second-order elliptic problems. *SIAM Journal on Numerical Analysis*, 41(6):2374–2399, 2003.

[42] J. Kautsky and N. K. Nichols. Equidistributing meshes with constraints. *SIAM J. SCI. STAT, COMPUT.*, 1(4):499–511, 1980.

[43] H. Kim and K. Nakahashi. Unstructured Adjoint Method for Navier-Stokes Equations. *JSME International Journal Series B*, 48:202–27, 2005.

[44] H. Kim, D. Sasaki, S. Obayashi, and K. Nakahashi. Aerodynamic Optimization of Supersonic Transport Wing Using Unstructured Adjoint Method. *AIAA Journal*, 39:1011–1020, 2001.

[45] H. L. Kline, Thomas D. Economon, and Juan J. Alonso. Multi-objective optimization of a hypersonic inlet using generalized outflow boundary conditions in the continuous adjoint method. In *54th AIAA Aerospace Sciences Meeting*, 2016.

[46] M. Kurzen, T. S. Phillips, and C. J. Roy. Method of nearby problems for generating exact solutions to 1D unsteady and 2D steady problems. Technical report, AIAA-2009-3652, 2009.

[47] M. G. Larson and T. J. Barth. A posteriori Error Estimation for Discontinuous Galerkin Approximations of Hyperbolic Systems. In *NAS Technical Report*, number NAS-99-010, 1999.

[48] Tobias Leicht and Ralf Hartmann. Anisotropic mesh refinement for discontinuous Galerkin methods in two-dimensional aerodynamic flow simulations. *International Journal for Numerical Methods in Fluids*, 56:2111–2138, 2008.

[49] B. Lindberg. Error estimation and iterative improvement for discretization algorithms. *BIT*, 20:486–500, 1980.

[50] C. A. Mavriplis. Adaptive mesh strategies for the spectral element method. *Computer Methods in Applied Mechanics and Engineering,*, 116(14), 1994.

[51] Fotis Mavriplis. CFD in aerospace in the new millenium. *Canadian Aeronautics and Space Journal*, 46(4):167–179, 2000.

[52] Christopher Michalak. *Efficient high-order accurate unstructured finite-volume algorithms for viscous and inviscid compressible flows*. PhD thesis, The University of British Columbia, Department of Mechanical Engineering, 2009.

[53] 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.

[54] Bijan Mohammadi and Olivier Pironneau. Shape Optimization in Fluid Mechanics. *Annu. Rev. Fluid Mech*, 36:11.1–11.25, 2004.

[55] A. Naumovich, M. Foerster, and R. Dwight. Algebraic multigrid within defect correction for the linearized euler equations. *Numerical Linear Algebra with Applications*, 17:307–324, 2009.

[56] Amir Nejat and Carl Ollivier-Gooch. A high-order accurate unstructured GMRES algorithm for inviscid compressible flows. In *Proceedings of the Seventeenth AIAA Computational Fluid Dynamics Conference*. American Institute of Aeronautics and Astronautics, June 2005.

[57] 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.

[58] Amir Nejat, Carl Ollivier-Gooch, and Christopher Michalak. Accuracy assessment methodology for a high-order unstructured finite volume solver. In *Proceedings of the Eighteenth AIAA Computational Fluid Dynamics Conference*, 2007.

[59] M. Nemec and M. J. Aftosmis. Adjoint error estimation and adaptive refinement for embedded-boundary cartesian meshes. In *Proceedings of the Eighteenth AIAA Computational Fluid Dynamics Conference*. American Institute of Aeronautics and Astronautics, 2007.

[60] Marian Nemec, Michael J. Aftosmis, and Mathias Wintzer. Adjoint-based adaptive mesh refinement for complex geometries. In *Forty-sixth AIAA Aerospace Sciences Meeting*, 2008. AIAA 2008-725.

[61] Hiroaki Nishikawa. Beyond interface gradient: A general principle for constructing diffusion scheme. In *Proceedings of the Fortieth AIAA Fluid Dynamics Conference and Exhibit*, 2010. AIAA paper 2010-5093.

[62] W. L. Oberkampf and C. J. Roy. Verification and Validation in Scientific Computing. *Cambridge University Press*, 2010.

[63] Carl Ollivier-Gooch. GRUMMP version 0.7.0 user's guide. Technical report, Department of Mechanical Engineering, The University of British Columbia, 2015.

[64] Carl Ollivier-Gooch, Amir Nejat, and Christopher Michalak. On obtaining high-order finite-volume solutions to the Euler equations on unstructured meshes. In *Proceedings of the Eighteenth AIAA Computational Fluid Dynamics Conference*, 2007. AIAA-2007-4464.

[65] Carl Ollivier-Gooch and Christopher Roy. Reducing truncation error on unstructured meshes by vertex movement. In *Proceedings of the Forty-Second AIAA Fluid Dynamics Conference*, 2012-2712. American Institute of Aeronautics and Astronautics, 2012.

[66] Carl F. Ollivier-Gooch and Michael Van Altena. A high-order accurate unstructured mesh finite-volume scheme for the advection-diffusion equation. *Journal of Computational Physics*, 181(2):729–752, 2002.

[67] Doug Pagnutti and Carl Ollivier-Gooch. A generalized framework for high order anisotropic mesh adaptation. *Computers and Structures*, 87:670–679, 2009.

[68] Michael Andrew Park. Adjoint-based, three-dimensional error prediction and grid adaptation. *AIAA Journal*, 42(9):1854–1862, 2004.

[69] V. Pereyra. On improving an approximate solution of a functional equation by deferred corrections. *Numer. Math.*, 8:376–391, 1966.

[70] V. Pereyra. Iterated deferred corrections for nonlinear boundary value problems. *Numer. Math*, 11:111–125, 1968.

[71] Tyrone Phillips, Christopher Roy, and Jeff Borggaard. Error Transport Equation Boundary Conditions for the Euler and Navier-Stokes Equations. In *52nd Aerospace Sciences Meeting, AIAA SciTech*, 2014-1432, 2014.

[72] Niles Pierce and Michael Giles. Adjoint recovery of superconvergent functionals from PDE approximations. *SIAM Review*, 42(2):247–264, June 2000.

[73] Niles Pierce and Michael Giles. Adjoint and defect error bounding and correction for functional estimates. *Journal of Computational Physics*, 200(2):769–794, 2004.

[74] O. Pironneau. On Optimum Design in Fluid Mechanics. *J. Fluid Mech.*, 64:97–110, 1974.

[75] Z. Qiao and X. Yang. Wing Design by Solving Adjoint Equations. In *40th AIAA Aerospace Sciences Meeting & Exhibit*, 2002-0263. American Institute of Aeronautics and Astronautics, 2002.

[76] Patrick J. Roache. Code verification by the method of manufactured solutions. *Journal of Fluids Engineering*, 124(1):4–10, March 2002.

[77] P. L. Roe. Characteristic-based schemes for the Euler equations. In *Annual Review of Fluid Mechanics*, volume 18, pages 337–365. Annuals Reviews, Inc., 1986.

[78] Phillip Roe. Error estimate for cell-vertex solutions od the compressible Eeler equations. Technical report, NASA Report, 1987.

[79] Christopher Roy. Strategies for driving mesh adaptation in CFD. In *Proceedings of the Forty-Seventh AIAA Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics, 2009.

[80] Christopher Roy. Review of discretization error estimators in scientific computing. In *Proceedings of the Forty-Eighth AIAA Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics, 2010. AIAA Paper 2010-0126.

[81] Mahkame Sharbatdar. Anisotropic mesh adaptation: Recovering quasi-structured meshes. Master's thesis, The University of British Columbia, Department of Mechanical Engineering, 2012.

[82] Mahkame Sharbatdar, Alireza Jalali, and Carl Ollivier-Gooch. Smoothed Truncation Error in Functional Error Estimation and Correction using Adjoint Methods

in an Unstructured Finite Volume Method. *Computers & Fluids*, 140:406–421, 2016.

[83] Mahkame Sharbatdar and Carl Ollivier-Gooch. Anisotropic mesh adaptation: Recovering quasi-structured meshes. In *Proceedings of the Fifty-First AIAA Aerospace Sciences Meeting*. AIAA 2013-0149, 2013.

[84] Mahkame Sharbatdar and Carl Ollivier-Gooch. Eigenanalysis of truncation and discretization error on unstructured meshes. In *21st AIAA Computational Fluid Dynamics Conference*, 2013-3089. American Institute of Aeronautics and Astronautics, 2013.

[85] Lei Shi and Z.J. Wang. Adjoint-Based Error Estimation and hp-Adaptation for the High-Order CPR Methods. In *51st AIAA Aerospace Science Meeting*, 2013.

[86] R. Skeel. A thoritical framework for proving accuracy results for deferred corrections. *SIAM J. Numer. Anal.*, 19:171–196, 1981.

[87] H.J. Stetter. The defect correction principle and discretization methods. *Numer. Math.*, 29:425–443, 1978.

[88] A. H. Stroud and Don Secrest. *Gaussian Quadrature Formulas*. Prentice-Hall, Englewood Cliffs, N.J., 1966.

[89] T. J. Tautges. CGM: A geometry interface for mesh generation, analysis and other applications. *Engineering with Computers*, 17(3):299–314, 2001.

[90] T.J. Tautges. The common geometry module (CGM): A generic, extensible geometry interface. In *Proceedings of the 9th International Meshing Roundtable, Sandia report SAND 2000-2207*, pages 337–359. Sandia National Laboratories, 2000.

[91] James L. Thomas, Boris Diskin, and Christopher Rumsey. Towards verification of unstructured-grid solvers. In *Forty-sixth AIAA Aerospace Sciences Meeting*, 2008. AIAA 2008-666.

[92] Michael Van Altena. High-order finite-volume discretisations for solving a modified advection-diffusion problem on unstructured triangular meshes. Master's thesis, The University of British Columbia, Department of Mechanical Engineering, October 1999.

[93] D. A. Venditti and D. L. Darmofal. Adjoint error estimation and grid adaptation for functional outputs: Application to quasi-one-dimensional flow. *Journal of Computational Physics*, 164(1):204–227, October 2000.

[94] D. A. Venditti and D. L. Darmofal. Grid adaptation for functional outputs: Application to two-dimensional inviscid flows. *Journal of Computational Physics*, 175(1):40–69, February 2002.

[95] D. A. Venditti and D. L. Darmofal. Grid adaptation for functional outputs: Application to two-dimensional viscous flows. *Journal of Computational Physics*, 187:22–46, 2003.

[96] N. J. Walkington. A C^1 tetrahedral finite element without edge degrees of freedom. *SIAM Journal of Numerical Analysis*, 52(1):330–342, 2014.

[97] L . Wang and D. Mavriplis. Adjoint-based h-p Adaptive Discontinuous Galerkin Methods for the Compressible Euler Equations. *Journal of Computational Physics*, 2009.

[98] M. Woopen, G. May, and J. Schutz. Adjoint-Based Error Estimation and Mesh Adaptation for Hybridized Discontinuous Galerkin Methods. *International Journal for numjerical methods in fluids*, 76:811–834, 2014.

[99] D. Zingg, S. De Rango, M. Nemec, and T. Pulliam. Comparison of several spatial discretizations for the Navier-Stokes equations. *Journal of Computational Physics*, 160:683–704, 2000.

Appendix A: Constraints Equations for Obtaining C^1 Interpolation of the Solution

Each basis function used for C^1 interpolation of the solution is a quintic polynomial:

$$\begin{aligned}
\varphi_i(\xi,\eta) &= c^i_{0,0} \\
&+ c^i_{1,0}\xi + c^i_{0,1}\eta \\
&+ c^i_{2,0}\xi^2 + c^i_{1,1}\xi\eta + c^i_{0,2}\eta^2 \\
&+ c^i_{3,0}\xi^3 + c^i_{2,1}\xi^2\eta + c^i_{1,2}\xi\eta^2 + c^i_{0,3}\eta^3 \\
&+ c^i_{4,0}\xi^4 + c^i_{3,1}\xi^3\eta + c^i_{2,2}\xi^2\eta^2 + c^i_{1,2}\xi\eta^3 + c^i_{0,4}\eta^4 \\
&+ c^i_{5,0}\xi^5 + c^i_{4,1}\xi^4\eta + c^i_{3,2}\xi^3\eta^2 + c^i_{2,3}\xi^2\eta^3 + c^i_{1,4}\xi\eta^4 + c^i_{0,5}\eta^5
\end{aligned}$$

The first derivatives are:

$$\begin{aligned}
\frac{\partial\varphi}{\partial\xi} &= c_{1,0} + 2c_{2,0}\xi + c_{1,1}\eta + 3c_{3,0}\xi^2 + 2c_{2,1}\xi\eta + c_{1,2}\eta^2 + 4c_{4,0}\xi^3 + 3c_{3,1}\xi^2\eta \\
&+ 2c_{2,2}\xi\eta^2 + c_{1,2}\eta^3 + 5c_{5,0}\xi^4 + 4c_{4,1}\xi^3\eta + 3c_{3,2}\xi^2\eta^2 + 2c_{2,3}\xi\eta^3 + c_{1,4}\eta^4
\end{aligned}$$

$$\begin{aligned}
\frac{\partial\varphi}{\partial\eta} &= c_{0,1} + c_{1,1}\xi + 2c_{0,2}\eta + c_{2,1}\xi^2 + 2c_{1,2}\xi\eta + 3c_{0,3}\eta^2 + c_{3,1}\xi^3 + 2c_{2,2}\xi^2\eta \\
&+ 3c_{1,2}\xi\eta^2 + 4c_{0,4}\eta^3 + c_{4,1}\xi^4 + 2c_{3,2}\xi^3\eta + 3c_{2,3}\xi^2\eta^2 + 4c_{1,4}\xi\eta^3 + 5c_{0,5}\eta^4
\end{aligned}$$

and the second derivatives are:

$$\begin{aligned}
\frac{\partial^2\varphi}{\partial\xi^2} &= 2c_{2,0} + 6c_{3,0}\xi + 2c_{2,1}\eta + 12c_{4,0}\xi^2 + 6c_{3,1}\xi\eta + \\
&\quad 2c_{2,2}\eta^2 + 20c_{5,0}\xi^3 + 12c_{4,1}\xi^2\eta + 6c_{3,2}\xi\eta^2 + 2c_{2,3}\eta^3 \\
\frac{\partial^2\varphi}{\partial\xi\partial\eta} &= c_{1,1} + 2c_{2,1}\xi + 2c_{1,2}\eta + 3c_{3,1}\xi^2 + 4c_{2,2}\xi\eta + \\
&\quad 3c_{1,2}\eta^2 + 4c_{4,1}\xi^3 + 6c_{3,2}\xi^2\eta + 6c_{2,3}\xi\eta^2 + 4c_{1,4}\eta^3 \\
\frac{\partial^2\varphi}{\partial\eta^2} &= 2c_{0,2} + 2c_{1,2}\xi + 6c_{0,3}\eta + 2c_{2,2}\xi^2 + 6c_{1,2}\xi\eta + \\
&\quad 12c_{0,4}\eta^2 + 2c_{3,2}\xi^3 + 6c_{2,3}\xi^2\eta + 12c_{1,4}\xi\eta^2 + 20c_{0,5}\eta^3
\end{aligned}$$

and the normal derivatives are:

$$\begin{aligned}
\frac{\partial\varphi}{\partial\xi}n_\xi + \frac{\partial\varphi}{\partial\eta}n_\eta &= (c_{1,0} + 2c_{2,0}\xi + c_{1,1}\eta + 3c_{3,0}\xi^2 + 2c_{2,1}\xi\eta + c_{1,2}\eta^2 + 4c_{4,0}\xi^3 + 3c_{3,1}\xi^2\eta + \\
&\quad 2c_{2,2}\xi\eta^2 + c_{1,2}\eta^3 + 5c_{5,0}\xi^4 + 4c_{4,1}\xi^3\eta + 3c_{3,2}\xi^2\eta^2 + 2c_{2,3}\xi\eta^3 + c_{1,4}\eta^4)n_\xi \\
&\quad (c_{0,1} + c_{1,1}\xi + 2c_{0,2}\eta + c_{2,1}\xi^2 + 2c_{1,2}\xi\eta + 3c_{0,3}\eta^2 + c_{3,1}\xi^3 + 2c_{2,2}\xi^2\eta + \\
&\quad 3c_{1,2}\xi\eta^2 + 4c_{0,4}\eta^3 + c_{4,1}\xi^4 + 2c_{3,2}\xi^3\eta + 3c_{2,3}\xi^2\eta^2 + 4c_{1,4}\xi\eta^3 + 5c_{0,5}\eta^4)n_\eta
\end{aligned}$$

The full list of 21 constraints for the 21 degrees of freedom, in Figure 3.8, which generate the full (21×21) system of equations is shown here. The constrains corresponding to φ_0:

$$\begin{aligned}
\varphi_0|_{(0,0)} &= c^0_{0,0} = 1 \\
\varphi_0|_{(1,0)} &= c^0_{0,0} + c^0_{1,0} + c^0_{2,0} + c^0_{3,0} + c^0_{4,0} + c^0_{5,0} = 0 \\
\varphi_0|_{(0,1)} &= c^0_{0,0} + c^0_{0,1} + c^0_{0,2} + c^0_{0,3} + c^0_{0,4} + c^0_{0,5} = 0
\end{aligned}$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_0}{\partial \xi}\right|_{(0,0)} &= c^0_{1,0} = 0 \\
\left.\frac{\partial \varphi_0}{\partial \xi}\right|_{(1,0)} &= c^0_{1,0} + 2c^0_{2,0} + 3c^0_{3,0} + 4c^0_{4,0} + 5c^0_{5,0} = 0 \\
\left.\frac{\partial \varphi_0}{\partial \xi}\right|_{(0,1)} &= c^0_{1,0} + c^0_{1,1}\eta + c^0_{1,2} + c^0_{1,2} + c^0_{1,4} = 0 \\
\left.\frac{\partial \varphi_0}{\partial \eta}\right|_{(0,0)} &= c^0_{0,1} = 0 \\
\left.\frac{\partial \varphi_0}{\partial \eta}\right|_{(1,0)} &= c^0_{0,1} + c^0_{1,1} + c^0_{2,1} + c^0_{3,1} + c^0_{4,1} = 0 \\
\left.\frac{\partial \varphi_0}{\partial \eta}\right|_{(0,1)} &= c^0_{0,1} + 2c^0_{0,2} + 3c^0_{0,3} + 4c^0_{0,4} + 5c^0_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_0}{\partial \xi^2}\right|_{(0,0)} &= 2c^0_{2,0} = 0 \\
\left.\frac{\partial^2 \varphi_0}{\partial \xi^2}\right|_{(1,0)} &= 2c^0_{2,0} + 6c^0_{3,0} + 12c^0_{4,0} + 20c^0_{5,0} = 0 \\
\left.\frac{\partial^2 \varphi_0}{\partial \xi^2}\right|_{(0,1)} &= 2c^0_{2,0} + 2c^0_{2,1} + 2c^0_{2,2} + 2c^0_{2,3} = 0 \\
\left.\frac{\partial^2 \varphi_0}{\partial \xi \partial \eta}\right|_{(0,0)} &= c^0_{1,1} = 0 \\
\left.\frac{\partial^2 \varphi_0}{\partial \xi \partial \eta}\right|_{(1,0)} &= c^0_{1,1} + 2c^0_{2,1} + 3c^0_{3,1} + 4c^0_{4,1} = 0 \\
\left.\frac{\partial^2 \varphi_0}{\partial \xi \partial \eta}\right|_{(0,1)} &= c^0_{1,1} + 2c^0_{1,2} + 3c^0_{1,2} + 4c^0_{1,4} = 0 \\
\left.\frac{\partial^2 \varphi_0}{\partial \eta^2}\right|_{(0,0)} &= 2c^0_{0,2} = 0 \\
\left.\frac{\partial^2 \varphi_0}{\partial \eta^2}\right|_{(1,0)} &= 2c^0_{0,2} + 2c^0_{1,2} + 2c^0_{2,2} + 2c^0_{3,2} = 0 \\
\left.\frac{\partial^2 \varphi_0}{\partial \eta^2}\right|_{(0,1)} &= 2c^0_{0,2} + 6c^0_{0,3} + 12c^0_{0,4} + 20c^0_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left(\frac{\partial\varphi_0}{\partial\xi}n_\xi+\frac{\partial\varphi_0}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_0}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^0_{0,1}+\frac{1}{2}c^0_{1,1}+\frac{1}{4}c^0_{2,1}+\frac{1}{8}c^0_{3,1}+\frac{1}{16}c^0_{4,1}=0\\
\left(\frac{\partial\varphi_0}{\partial\xi}n_\xi+\frac{\partial\varphi_0}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^0_{1,0}+c^0_{2,0}+c^0_{1,1}+\frac{3}{4}c^0_{3,0}+\frac{3}{4}c^0_{2,1}+\frac{3}{4}c^0_{1,2}+\frac{1}{2}c^0_{4,0}+\\
&\quad \frac{1}{2}c^0_{3,1}+\frac{1}{2}c^0_{2,2}+\frac{1}{2}c^0_{1,2}+\frac{5}{16}c^0_{5,0}+\frac{9}{16}c^0_{4,1}+\frac{7}{16}c^0_{3,2}+\\
&\quad \frac{7}{16}c^0_{2,3}+\frac{5}{16}c^0_{1,4}+c^0_{0,1}+c^0_{0,2}+\frac{3}{4}c^0_{0,3}+\frac{1}{2}c^0_{0,4}+\frac{5}{16}c^0_{0,5})=0\\
\left(\frac{\partial\varphi_0}{\partial\xi}n_\xi+\frac{\partial\varphi_0}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_0}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^0_{0,1}+c^0_{0,2}+\frac{3}{4}c^0_{0,3}+\frac{1}{2}c^0_{0,4}+\frac{5}{16}c^0_{0,5}\right)=0
\end{aligned}
$$

The constrains corresponding to φ_1:

$$
\begin{aligned}
\varphi_1|_{(0,0)} &= c^1_{0,0}=0\\
\varphi_1|_{(1,0)} &= c^1_{0,0}+c^1_{1,0}+c^1_{2,0}+c^1_{3,0}+c^1_{4,0}+c^1_{5,0}=0\\
\varphi_1|_{(0,1)} &= c^1_{0,0}+c^1_{0,1}+c^1_{0,2}+c^1_{0,3}+c^1_{0,4}+c^1_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial\varphi_1}{\partial\xi}\right|_{(0,0)} &= c^1_{1,0}=1\\
\left.\frac{\partial\varphi_1}{\partial\xi}\right|_{(1,0)} &= c^1_{1,0}+2c^1_{2,0}+3c^1_{3,0}+4c^1_{4,0}+5c^1_{5,0}=0\\
\left.\frac{\partial\varphi_1}{\partial\xi}\right|_{(0,1)} &= c^1_{1,0}+c^1_{1,1}\eta+c^1_{1,2}+c^1_{1,2}+c^1_{1,4}=0\\
\left.\frac{\partial\varphi_1}{\partial\eta}\right|_{(0,0)} &= c^1_{0,1}=0\\
\left.\frac{\partial\varphi_1}{\partial\eta}\right|_{(1,0)} &= c^1_{0,1}+c^1_{1,1}+c^1_{2,1}+c^1_{3,1}+c^1_{4,1}=0\\
\left.\frac{\partial\varphi_1}{\partial\eta}\right|_{(0,1)} &= c^1_{0,1}+2c^1_{0,2}+3c^1_{0,3}+4c^1_{0,4}+5c^1_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2\varphi_1}{\partial\xi^2}\right|_{(0,0)} &= 2c^1_{2,0} = 0\\
\left.\frac{\partial^2\varphi_1}{\partial\xi^2}\right|_{(1,0)} &= 2c^1_{2,0} + 6c^1_{3,0} + 12c^1_{4,0} + 20c^1_{5,0} = 0\\
\left.\frac{\partial^2\varphi_1}{\partial\xi^2}\right|_{(0,1)} &= 2c^1_{2,0} + 2c^1_{2,1} + 2c^1_{2,2} + 2c^1_{2,3} = 0\\
\left.\frac{\partial^2\varphi_1}{\partial\xi\partial\eta}\right|_{(0,0)} &= c^1_{1,1} = 0\\
\left.\frac{\partial^2\varphi_1}{\partial\xi\partial\eta}\right|_{(1,0)} &= c^1_{1,1} + 2c^1_{2,1} + 3c^1_{3,1} + 4c^1_{4,1} = 0\\
\left.\frac{\partial^2\varphi_1}{\partial\xi\partial\eta}\right|_{(0,1)} &= c^1_{1,1} + 2c^1_{1,2} + 3c^1_{1,2} + 4c^1_{1,4} = 0\\
\left.\frac{\partial^2\varphi_1}{\partial\eta^2}\right|_{(0,0)} &= 2c^1_{0,2} = 0\\
\left.\frac{\partial^2\varphi_1}{\partial\eta^2}\right|_{(1,0)} &= 2c^1_{0,2} + 2c^1_{1,2} + 2c^1_{2,2} + 2c^1_{3,2} = 0\\
\left.\frac{\partial^2\varphi_1}{\partial\eta^2}\right|_{(0,1)} &= 2c^1_{0,2} + 6c^1_{0,3} + 12c^1_{0,4} + 20c^1_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\left(\frac{\partial\varphi_1}{\partial\xi}n_\xi + \frac{\partial\varphi_1}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_1}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^1_{0,1} + \frac{1}{2}c^1_{1,1} + \frac{1}{4}c^1_{2,1} + \frac{1}{8}c^1_{3,1} + \frac{1}{16}c^1_{4,1} = 0\\
\left.\left(\frac{\partial\varphi_1}{\partial\xi}n_\xi + \frac{\partial\varphi_1}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^1_{1,0} + c^1_{2,0} + c^1_{1,1} + \frac{3}{4}c^1_{3,0} + \frac{3}{4}c^1_{2,1} + \frac{3}{4}c^1_{1,2} + \frac{1}{2}c^1_{4,0} +\\
&\quad \frac{1}{2}c^1_{3,1} + \frac{1}{2}c^1_{2,2} + \frac{1}{2}c^1_{1,2} + \frac{5}{16}c^1_{5,0} + \frac{9}{16}c^1_{4,1} + \frac{7}{16}c^1_{3,2} +\\
&\quad \frac{7}{16}c^1_{2,3} + \frac{5}{16}c^1_{1,4} + c^1_{0,1} + c^1_{0,2} + \frac{3}{4}c^1_{0,3} + \frac{1}{2}c^1_{0,4} + \frac{5}{16}c^1_{0,5}) = 0\\
\left.\left(\frac{\partial\varphi_1}{\partial\xi}n_\xi + \frac{\partial\varphi_1}{\partial\eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_1}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^1_{0,1} + c^1_{0,2} + \frac{3}{4}c^1_{0,3} + \frac{1}{2}c^1_{0,4} + \frac{5}{16}c^1_{0,5}\right) = 0
\end{aligned}
$$

The constrains corresponding to φ_2:

$$
\begin{aligned}
\varphi_2|_{(0,0)} &= c^2_{0,0} = 0\\
\varphi_2|_{(1,0)} &= c^2_{0,0} + c^2_{1,0} + c^2_{2,0} + c^2_{3,0} + c^2_{4,0} + c^2_{5,0} = 0\\
\varphi_2|_{(0,1)} &= c^2_{0,0} + c^2_{0,1} + c^2_{0,2} + c^2_{0,3} + c^2_{0,4} + c^2_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_2}{\partial \xi}\right|_{(0,0)} &= c_{1,0}^2 = 0 \\
\left.\frac{\partial \varphi_2}{\partial \xi}\right|_{(1,0)} &= c_{1,0}^2 + 2c_{2,0}^2 + 3c_{3,0}^2 + 4c_{4,0}^2 + 5c_{5,0}^2 = 0 \\
\left.\frac{\partial \varphi_2}{\partial \xi}\right|_{(0,1)} &= c_{1,0}^2 + c_{1,1}^2\eta + c_{1,2}^2 + c_{1,2}^2 + c_{1,4}^2 = 0 \\
\left.\frac{\partial \varphi_2}{\partial \eta}\right|_{(0,0)} &= c_{0,1}^2 = 1 \\
\left.\frac{\partial \varphi_2}{\partial \eta}\right|_{(1,0)} &= c_{0,1}^2 + c_{1,1}^2 + c_{2,1}^2 + c_{3,1}^2 + c_{4,1}^2 = 0 \\
\left.\frac{\partial \varphi_2}{\partial \eta}\right|_{(0,1)} &= c_{0,1}^2 + 2c_{0,2}^2 + 3c_{0,3}^2 + 4c_{0,4}^2 + 5c_{0,5}^2 = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_2}{\partial \xi^2}\right|_{(0,0)} &= 2c_{2,0}^2 = 0 \\
\left.\frac{\partial^2 \varphi_2}{\partial \xi^2}\right|_{(1,0)} &= 2c_{2,0}^2 + 6c_{3,0}^2 + 12c_{4,0}^2 + 20c_{5,0}^2 = 0 \\
\left.\frac{\partial^2 \varphi_2}{\partial \xi^2}\right|_{(0,1)} &= 2c_{2,0}^2 + 2c_{2,1}^2 + 2c_{2,2}^2 + 2c_{2,3}^2 = 0 \\
\left.\frac{\partial^2 \varphi_2}{\partial \xi \partial \eta}\right|_{(0,0)} &= c_{1,1}^2 = 0 \\
\left.\frac{\partial^2 \varphi_2}{\partial \xi \partial \eta}\right|_{(1,0)} &= c_{1,1}^2 + 2c_{2,1}^2 + 3c_{3,1}^2 + 4c_{4,1}^2 = 0 \\
\left.\frac{\partial^2 \varphi_2}{\partial \xi \partial \eta}\right|_{(0,1)} &= c_{1,1}^2 + 2c_{1,2}^2 + 3c_{1,2}^2 + 4c_{1,4}^2 = 0 \\
\left.\frac{\partial^2 \varphi_2}{\partial \eta^2}\right|_{(0,0)} &= 2c_{0,2}^2 = 0 \\
\left.\frac{\partial^2 \varphi_2}{\partial \eta^2}\right|_{(1,0)} &= 2c_{0,2}^2 + 2c_{1,2}^2 + 2c_{2,2}^2 + 2c_{3,2}^2 = 0 \\
\left.\frac{\partial^2 \varphi_2}{\partial \eta^2}\right|_{(0,1)} &= 2c_{0,2}^2 + 6c_{0,3}^2 + 12c_{0,4}^2 + 20c_{0,5}^2 = 0
\end{aligned}
$$

$$
\begin{aligned}
\left(\frac{\partial\varphi_2}{\partial\xi}n_\xi+\frac{\partial\varphi_2}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_2}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c_{0,1}^2+\frac{1}{2}c_{1,1}^2+\frac{1}{4}c_{2,1}^2+\frac{1}{8}c_{3,1}^2+\frac{1}{16}c_{4,1}^2=0\\
\left(\frac{\partial\varphi_2}{\partial\xi}n_\xi+\frac{\partial\varphi_2}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c_{1,0}^2+c_{2,0}^2+c_{1,1}^2+\frac{3}{4}c_{3,0}^2+\frac{3}{4}c_{2,1}^2+\frac{3}{4}c_{1,2}^2+\frac{1}{2}c_{4,0}^2+\\
&\quad \frac{1}{2}c_{3,1}^2+\frac{1}{2}c_{2,2}^2+\frac{1}{2}c_{1,2}^2+\frac{5}{16}c_{5,0}^2+\frac{9}{16}c_{4,1}^{20}+\frac{7}{16}c_{3,2}^2+\\
&\quad \frac{7}{16}c_{2,3}^2+\frac{5}{16}c_{1,4}^2+c_{0,1}^2+c_{0,2}^2+\frac{3}{4}c_{0,3}^2+\frac{1}{2}c_{0,4}^2+\frac{5}{16}c_{0,5}^2)=0\\
\left(\frac{\partial\varphi_2}{\partial\xi}n_\xi+\frac{\partial\varphi_2}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_2}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c_{0,1}^2+c_{0,2}^2+\frac{3}{4}c_{0,3}^2+\frac{1}{2}c_{0,4}^2+\frac{5}{16}c_{0,5}^2\right)=0
\end{aligned}
$$

The constrains corresponding to φ_3:

$$
\begin{aligned}
\varphi_3|_{(0,0)} &= c_{0,0}^3=0\\
\varphi_3|_{(1,0)} &= c_{0,0}^0+c_{1,0}^0+c_{2,0}^0+c_{3,0}^0+c_{4,0}^0+c_{5,0}^0=0\\
\varphi_3|_{(0,1)} &= c_{0,0}^0+c_{0,1}^0+c_{0,2}^0+c_{0,3}^0+c_{0,4}^0+c_{0,5}^0=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial\varphi_3}{\partial\xi}\right|_{(0,0)} &= c_{1,0}^3=0\\
\left.\frac{\partial\varphi_3}{\partial\xi}\right|_{(1,0)} &= c_{1,0}^3+2c_{2,0}^3+3c_{3,0}^3+4c_{4,0}^3+5c_{5,0}^3=0\\
\left.\frac{\partial\varphi_3}{\partial\xi}\right|_{(0,1)} &= c_{1,0}^3+c_{1,1}^3\eta+c_{1,2}^3+c_{1,2}^3+c_{1,4}^3=0\\
\left.\frac{\partial\varphi_3}{\partial\eta}\right|_{(0,0)} &= c_{0,1}^3=0\\
\left.\frac{\partial\varphi_3}{\partial\eta}\right|_{(1,0)} &= c_{0,1}^3+c_{1,1}^3+c_{2,1}^3+c_{3,1}^3+c_{4,1}^3=0\\
\left.\frac{\partial\varphi_3}{\partial\eta}\right|_{(0,1)} &= c_{0,1}^3+2c_{0,2}^3+3c_{0,3}^3+4c_{0,4}^3+5c_{0,5}^3=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2\varphi_3}{\partial\xi^2}\right|_{(0,0)} &= 2c^3_{2,0} = 1\\
\left.\frac{\partial^2\varphi_3}{\partial\xi^2}\right|_{(1,0)} &= 2c^3_{2,0} + 6c^3_{3,0} + 12c^3_{4,0} + 20c^3_{5,0} = 0\\
\left.\frac{\partial^2\varphi_3}{\partial\xi^2}\right|_{(0,1)} &= 2c^3_{2,0} + 2c^3_{2,1} + 2c^3_{2,2} + 2c^3_{2,3} = 0\\
\left.\frac{\partial^2\varphi_3}{\partial\xi\partial\eta}\right|_{(0,0)} &= c^3_{1,1} = 0\\
\left.\frac{\partial^2\varphi_3}{\partial\xi\partial\eta}\right|_{(1,0)} &= c^3_{1,1} + 2c^3_{2,1} + 3c^3_{3,1} + 4c^3_{4,1} = 0\\
\left.\frac{\partial^2\varphi_3}{\partial\xi\partial\eta}\right|_{(0,1)} &= c^3_{1,1} + 2c^3_{1,2} + 3c^3_{1,2} + 4c^3_{1,4} = 0\\
\left.\frac{\partial^2\varphi_3}{\partial\eta^2}\right|_{(0,0)} &= 2c^3_{0,2} = 0\\
\left.\frac{\partial^2\varphi_0}{\partial\eta^2}\right|_{(1,0)} &= 2c^3_{0,2} + 2c^3_{1,2} + 2c^3_{2,2} + 2c^3_{3,2} = 0\\
\left.\frac{\partial^2\varphi_3}{\partial\eta^2}\right|_{(0,1)} &= 2c^3_{0,2} + 6c^3_{0,3} + 12c^3_{0,4} + 20c^3_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\left(\frac{\partial\varphi_3}{\partial\xi}n_\xi + \frac{\partial\varphi_3}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_3}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^3_{0,1} + \frac{1}{2}c^3_{1,1} + \frac{1}{4}c^3_{2,1} + \frac{1}{8}c^3_{3,1} + \frac{1}{16}c^3_{4,1} = 0\\
\left.\left(\frac{\partial\varphi_3}{\partial\xi}n_\xi + \frac{\partial\varphi_3}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^3_{1,0} + c^3_{2,0} + c^3_{1,1} + \frac{3}{4}c^3_{3,0} + \frac{3}{4}c^3_{2,1} + \frac{3}{4}c^3_{1,2} + \frac{1}{2}c^3_{4,0} +\\
&\quad \frac{1}{2}c^3_{3,1} + \frac{1}{2}c^3_{2,2} + \frac{1}{2}c^3_{1,2} + \frac{5}{16}c^3_{5,0} + \frac{9}{16}c^3_{4,1} + \frac{7}{16}c^3_{3,2} +\\
&\quad \frac{7}{16}c^3_{2,3} + \frac{5}{16}c^3_{1,4} + c^3_{0,1} + c^3_{0,2} + \frac{3}{4}c^3_{0,3} + \frac{1}{2}c^3_{0,4} + \frac{5}{16}c^3_{0,5}) = 0\\
\left.\left(\frac{\partial\varphi_3}{\partial\xi}n_\xi + \frac{\partial\varphi_3}{\partial\eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_3}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^3_{0,1} + c^3_{0,2} + \frac{3}{4}c^3_{0,3} + \frac{1}{2}c^3_{0,4} + \frac{5}{16}c^3_{0,5}\right) = 0
\end{aligned}
$$

The constrains corresponding to φ_4:

$$
\begin{aligned}
\varphi_4|_{(0,0)} &= c^4_{0,0} = 0\\
\varphi_4|_{(1,0)} &= c^4_{0,0} + c^4_{1,0} + c^4_{2,0} + c^4_{3,0} + c^4_{4,0} + c^4_{5,0} = 0\\
\varphi_4|_{(0,1)} &= c^4_{0,0} + c^4_{0,1} + c^4_{0,2} + c^4_{0,3} + c^4_{0,4} + c^4_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_4}{\partial \xi}\right|_{(0,0)} &= c^4_{1,0} = 0 \\
\left.\frac{\partial \varphi_4}{\partial \xi}\right|_{(1,0)} &= c^4_{1,0} + 2c^4_{2,0} + 3c^4_{3,0} + 4c^4_{4,0} + 5c^4_{5,0} = 0 \\
\left.\frac{\partial \varphi_4}{\partial \xi}\right|_{(0,1)} &= c^4_{1,0} + c^4_{1,1}\eta + c^4_{1,2} + c^4_{1,2} + c^4_{1,4} = 0 \\
\left.\frac{\partial \varphi_4}{\partial \eta}\right|_{(0,0)} &= c^4_{0,1} = 0 \\
\left.\frac{\partial \varphi_4}{\partial \eta}\right|_{(1,0)} &= c^4_{0,1} + c^4_{1,1} + c^4_{2,1} + c^4_{3,1} + c^4_{4,1} = 0 \\
\left.\frac{\partial \varphi_4}{\partial \eta}\right|_{(0,1)} &= c^4_{0,1} + 2c^4_{0,2} + 3c^4_{0,3} + 4c^4_{0,4} + 5c^4_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_4}{\partial \xi^2}\right|_{(0,0)} &= 2c^4_{2,0} = 0 \\
\left.\frac{\partial^2 \varphi_4}{\partial \xi^2}\right|_{(1,0)} &= 2c^4_{2,0} + 6c^4_{3,0} + 12c^4_{4,0} + 20c^4_{5,0} = 0 \\
\left.\frac{\partial^2 \varphi_4}{\partial \xi^2}\right|_{(0,1)} &= 2c^4_{2,0} + 2c^4_{2,1} + 2c^4_{2,2} + 2c^4_{2,3} = 0 \\
\left.\frac{\partial^2 \varphi_4}{\partial \xi \partial \eta}\right|_{(0,0)} &= c^4_{1,1} = 1 \\
\left.\frac{\partial^2 \varphi_4}{\partial \xi \partial \eta}\right|_{(1,0)} &= c^4_{1,1} + 2c^4_{2,1} + 3c^4_{3,1} + 4c^4_{4,1} = 0 \\
\left.\frac{\partial^2 \varphi_0}{\partial \xi \partial \eta}\right|_{(0,1)} &= c^4_{1,1} + 2c^4_{1,2} + 3c^4_{1,2} + 4c^4_{1,4} = 0 \\
\left.\frac{\partial^2 \varphi_4}{\partial \eta^2}\right|_{(0,0)} &= 2c^4_{0,2} = 0 \\
\left.\frac{\partial^2 \varphi_4}{\partial \eta^2}\right|_{(1,0)} &= 2c^4_{0,2} + 2c^4_{1,2} + 2c^4_{2,2} + 2c^4_{3,2} = 0 \\
\left.\frac{\partial^2 \varphi_4}{\partial \eta^2}\right|_{(0,1)} &= 2c^4_{0,2} + 6c^4_{0,3} + 12c^4_{0,4} + 20c^4_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left(\frac{\partial\varphi_4}{\partial\xi}n_\xi+\frac{\partial\varphi_4}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_4}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^4_{0,1}+\frac{1}{2}c^4_{1,1}+\frac{1}{4}c^4_{2,1}+\frac{1}{8}c^4_{3,1}+\frac{1}{16}c^4_{4,1}=0\\
\left(\frac{\partial\varphi_4}{\partial\xi}n_\xi+\frac{\partial\varphi_4}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^4_{1,0}+c^4_{2,0}+c^4_{1,1}+\frac{3}{4}c^4_{3,0}+\frac{3}{4}c^4_{2,1}+\frac{3}{4}c^4_{1,2}+\frac{1}{2}c^4_{4,0}+\\
&\quad \frac{1}{2}c^4_{3,1}+\frac{1}{2}c^4_{2,2}+\frac{1}{2}c^4_{1,2}+\frac{5}{16}c^4_{5,0}+\frac{9}{16}c^4_{4,1}+\frac{7}{16}c^4_{3,2}+\\
&\quad \frac{7}{16}c^4_{2,3}+\frac{5}{16}c^4_{1,4}+c^4_{0,1}+c^4_{0,2}+\frac{3}{4}c^4_{0,3}+\frac{1}{2}c^4_{0,4}+\frac{5}{16}c^4_{0,5})=0\\
\left(\frac{\partial\varphi_4}{\partial\xi}n_\xi+\frac{\partial\varphi_4}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_4}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^4_{0,1}+c^4_{0,2}+\frac{3}{4}c^4_{0,3}+\frac{1}{2}c^4_{0,4}+\frac{5}{16}c^4_{0,5}\right)=0
\end{aligned}
$$

The constrains corresponding to φ_5:

$$
\begin{aligned}
\varphi_5|_{(0,0)} &= c^5_{0,0}=0\\
\varphi_5|_{(1,0)} &= c^5_{0,0}+c^5_{1,0}+c^5_{2,0}+c^5_{3,0}+c^5_{4,0}+c^5_{5,0}=0\\
\varphi_5|_{(0,1)} &= c^5_{0,0}+c^5_{0,1}+c^5_{0,2}+c^5_{0,3}+c^5_{0,4}+c^5_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial\varphi_5}{\partial\xi}\right|_{(0,0)} &= c^5_{1,0}=0\\
\left.\frac{\partial\varphi_5}{\partial\xi}\right|_{(1,0)} &= c^5_{1,0}+2c^5_{2,0}+3c^5_{3,0}+4c^5_{4,0}+5c^5_{5,0}=0\\
\left.\frac{\partial\varphi_5}{\partial\xi}\right|_{(0,1)} &= c^5_{1,0}+c^5_{1,1}\eta+c^5_{1,2}+c^5_{1,2}+c^5_{1,4}=0\\
\left.\frac{\partial\varphi_5}{\partial\eta}\right|_{(0,0)} &= c^5_{0,1}=0\\
\left.\frac{\partial\varphi_5}{\partial\eta}\right|_{(1,0)} &= c^5_{0,1}+c^5_{1,1}+c^5_{2,1}+c^5_{3,1}+c^5_{4,1}=0\\
\left.\frac{\partial\varphi_5}{\partial\eta}\right|_{(0,1)} &= c^5_{0,1}+2c^5_{0,2}+3c^5_{0,3}+4c^5_{0,4}+5c^5_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2\varphi_5}{\partial\xi^2}\right|_{(0,0)} &= 2c^5_{2,0} = 0\\
\left.\frac{\partial^2\varphi_5}{\partial\xi^2}\right|_{(1,0)} &= 2c^5_{2,0} + 6c^5_{3,0} + 12c^5_{4,0} + 20c^5_{5,0} = 0\\
\left.\frac{\partial^2\varphi_5}{\partial\xi^2}\right|_{(0,1)} &= 2c^5_{2,0} + 2c^5_{2,1} + 2c^5_{2,2} + 2c^5_{2,3} = 0\\
\left.\frac{\partial^2\varphi_5}{\partial\xi\partial\eta}\right|_{(0,0)} &= c^5_{1,1} = 0\\
\left.\frac{\partial^2\varphi_5}{\partial\xi\partial\eta}\right|_{(1,0)} &= c^5_{1,1} + 2c^5_{2,1} + 3c^5_{3,1} + 4c^5_{4,1} = 0\\
\left.\frac{\partial^2\varphi_5}{\partial\xi\partial\eta}\right|_{(0,1)} &= c^5_{1,1} + 2c^5_{1,2} + 3c^5_{1,2} + 4c^5_{1,4} = 0\\
\left.\frac{\partial^2\varphi_5}{\partial\eta^2}\right|_{(0,0)} &= 2c^5_{0,2} = 1\\
\left.\frac{\partial^2\varphi_5}{\partial\eta^2}\right|_{(1,0)} &= 2c^5_{0,2} + 2c^5_{1,2} + 2c^5_{2,2} + 2c^5_{3,2} = 0\\
\left.\frac{\partial^2\varphi_5}{\partial\eta^2}\right|_{(0,1)} &= 2c^5_{0,2} + 6c^5_{0,3} + 12c^5_{0,4} + 20c^5_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\left(\frac{\partial\varphi_5}{\partial\xi}n_\xi + \frac{\partial\varphi_5}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_5}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^5_{0,1} + \frac{1}{2}c^5_{1,1} + \frac{1}{4}c^5_{2,1} + \frac{1}{8}c^5_{3,1} + \frac{1}{16}c^5_{4,1} = 0\\
\left.\left(\frac{\partial\varphi_5}{\partial\xi}n_\xi + \frac{\partial\varphi_5}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^5_{1,0} + c^5_{2,0} + c^5_{1,1} + \frac{3}{4}c^5_{3,0} + \frac{3}{4}c^5_{2,1} + \frac{3}{4}c^5_{1,2} + \frac{1}{2}c^5_{4,0} +\\
&\quad \frac{1}{2}c^5_{3,1} + \frac{1}{2}c^5_{2,2} + \frac{1}{2}c^5_{1,2} + \frac{5}{16}c^5_{5,0} + \frac{9}{16}c^5_{4,1} + \frac{7}{16}c^5_{3,2} +\\
&\quad \frac{7}{16}c^5_{2,3} + \frac{5}{16}c^5_{1,4} + c^5_{0,1} + c^5_{0,2} + \frac{3}{4}c^5_{0,3} + \frac{1}{2}c^5_{0,4} + \frac{5}{16}c^5_{0,5}) = 0\\
\left.\left(\frac{\partial\varphi_5}{\partial\xi}n_\xi + \frac{\partial\varphi_5}{\partial\eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_5}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^5_{0,1} + c^5_{0,2} + \frac{3}{4}c^5_{0,3} + \frac{1}{2}c^5_{0,4} + \frac{5}{16}c^5_{0,5}\right) = 0
\end{aligned}
$$

The constrains corresponding to φ_6:

$$
\begin{aligned}
\varphi_6|_{(0,0)} &= c^6_{0,0} = 0\\
\varphi_6|_{(1,0)} &= c^6_{0,0} + c^6_{1,0} + c^6_{2,0} + c^6_{3,0} + c^6_{4,0} + c^6_{5,0} = 1\\
\varphi_6|_{(0,1)} &= c^6_{0,0} + c^6_{0,1} + c^6_{0,2} + c^6_{0,3} + c^6_{0,4} + c^6_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_6}{\partial \xi}\right|_{(0,0)} &= c^6_{1,0} = 0 \\
\left.\frac{\partial \varphi_6}{\partial \xi}\right|_{(1,0)} &= c^6_{1,0} + 2c^6_{2,0} + 3c^6_{3,0} + 4c^6_{4,0} + 5c^6_{5,0} = 0 \\
\left.\frac{\partial \varphi_6}{\partial \xi}\right|_{(0,1)} &= c^6_{1,0} + c^6_{1,1}\eta + c^6_{1,2} + c^6_{1,2} + c^6_{1,4} = 0 \\
\left.\frac{\partial \varphi_6}{\partial \eta}\right|_{(0,0)} &= c^6_{0,1} = 0 \\
\left.\frac{\partial \varphi_6}{\partial \eta}\right|_{(1,0)} &= c^6_{0,1} + c^6_{1,1} + c^6_{2,1} + c^6_{3,1} + c^6_{4,1} = 0 \\
\left.\frac{\partial \varphi_6}{\partial \eta}\right|_{(0,1)} &= c^6_{0,1} + 2c^6_{0,2} + 3c^6_{0,3} + 4c^6_{0,4} + 5c^6_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_6}{\partial \xi^2}\right|_{(0,0)} &= 2c^6_{2,0} = 0 \\
\left.\frac{\partial^2 \varphi_6}{\partial \xi^2}\right|_{(1,0)} &= 2c^6_{2,0} + 6c^6_{3,0} + 12c^6_{4,0} + 20c^6_{5,0} = 0 \\
\left.\frac{\partial^2 \varphi_6}{\partial \xi^2}\right|_{(0,1)} &= 2c^6_{2,0} + 2c^6_{2,1} + 2c^6_{2,2} + 2c^6_{2,3} = 0 \\
\left.\frac{\partial^2 \varphi_6}{\partial \xi \partial \eta}\right|_{(0,0)} &= c^6_{1,1} = 0 \\
\left.\frac{\partial^2 \varphi_6}{\partial \xi \partial \eta}\right|_{(1,0)} &= c^6_{1,1} + 2c^6_{2,1} + 3c^6_{3,1} + 4c^6_{4,1} = 0 \\
\left.\frac{\partial^2 \varphi_6}{\partial \xi \partial \eta}\right|_{(0,1)} &= c^6_{1,1} + 2c^6_{1,2} + 3c^6_{1,2} + 4c^6_{1,4} = 0 \\
\left.\frac{\partial^2 \varphi_6}{\partial \eta^2}\right|_{(0,0)} &= 2c^6_{0,2} = 0 \\
\left.\frac{\partial^2 \varphi_6}{\partial \eta^2}\right|_{(1,0)} &= 2c^6_{0,2} + 2c^6_{1,2} + 2c^6_{2,2} + 2c^6_{3,2} = 0 \\
\left.\frac{\partial^2 \varphi_6}{\partial \eta^2}\right|_{(0,1)} &= 2c^6_{0,2} + 6c^6_{0,3} + 12c^6_{0,4} + 20c^6_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left(\frac{\partial\varphi_6}{\partial\xi}n_\xi+\frac{\partial\varphi_6}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_6}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^6_{0,1}+\frac{1}{2}c^6_{1,1}+\frac{1}{4}c^6_{2,1}+\frac{1}{8}c^6_{3,1}+\frac{1}{16}c^6_{4,1}=0\\
\left(\frac{\partial\varphi_6}{\partial\xi}n_\xi+\frac{\partial\varphi_6}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^6_{1,0}+c^6_{2,0}+c^6_{1,1}+\frac{3}{4}c^6_{3,0}+\frac{3}{4}c^6_{2,1}+\frac{3}{4}c^6_{1,2}+\frac{1}{2}c^6_{4,0}+\\
&\quad \frac{1}{2}c^6_{3,1}+\frac{1}{2}c^6_{2,2}+\frac{1}{2}c^6_{1,2}+\frac{5}{16}c^6_{5,0}+\frac{9}{16}c^6_{4,1}+\frac{7}{16}c^6_{3,2}+\\
&\quad \frac{7}{16}c^6_{2,3}+\frac{5}{16}c^6_{1,4}+c^6_{0,1}+c^6_{0,2}+\frac{3}{4}c^6_{0,3}+\frac{1}{2}c^6_{0,4}+\frac{5}{16}c^6_{0,5})=0\\
\left(\frac{\partial\varphi_6}{\partial\xi}n_\xi+\frac{\partial\varphi_6}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_6}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^6_{0,1}+c^6_{0,2}+\frac{3}{4}c^6_{0,3}+\frac{1}{2}c^6_{0,4}+\frac{5}{16}c^6_{0,5}\right)=0
\end{aligned}
$$

The constrains corresponding to φ_7:

$$
\begin{aligned}
\varphi_7|_{(0,0)} &= c^7_{0,0}=0\\
\varphi_7|_{(1,0)} &= c^7_{0,0}+c^7_{1,0}+c^7_{2,0}+c^7_{3,0}+c^7_{4,0}+c^7_{5,0}=0\\
\varphi_7|_{(0,1)} &= c^7_{0,0}+c^7_{0,1}+c^7_{0,2}+c^7_{0,3}+c^7_{0,4}+c^7_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial\varphi_7}{\partial\xi}\right|_{(0,0)} &= c^7_{1,0}=0\\
\left.\frac{\partial\varphi_7}{\partial\xi}\right|_{(1,0)} &= c^7_{1,0}+2c^7_{2,0}+3c^7_{3,0}+4c^7_{4,0}+5c^7_{5,0}=1\\
\left.\frac{\partial\varphi_7}{\partial\xi}\right|_{(0,1)} &= c^7_{1,0}+c^7_{1,1}\eta+c^7_{1,2}+c^7_{1,2}+c^7_{1,4}=0\\
\left.\frac{\partial\varphi_7}{\partial\eta}\right|_{(0,0)} &= c^7_{0,1}=0\\
\left.\frac{\partial\varphi_7}{\partial\eta}\right|_{(1,0)} &= c^7_{0,1}+c^7_{1,1}+c^7_{2,1}+c^7_{3,1}+c^7_{4,1}=0\\
\left.\frac{\partial\varphi_7}{\partial\eta}\right|_{(0,1)} &= c^7_{0,1}+2c^7_{0,2}+3c^7_{0,3}+4c^7_{0,4}+5c^7_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2\varphi_7}{\partial\xi^2}\right|_{(0,0)} &= 2c^7_{2,0} = 0\\
\left.\frac{\partial^2\varphi_7}{\partial\xi^2}\right|_{(1,0)} &= 2c^7_{2,0} + 6c^7_{3,0} + 12c^7_{4,0} + 20c^7_{5,0} = 0\\
\left.\frac{\partial^2\varphi_7}{\partial\xi^2}\right|_{(0,1)} &= 2c^7_{2,0} + 2c^7_{2,1} + 2c^7_{2,2} + 2c^7_{2,3} = 0\\
\left.\frac{\partial^2\varphi_7}{\partial\xi\partial\eta}\right|_{(0,0)} &= c^7_{1,1} = 0\\
\left.\frac{\partial^2\varphi_7}{\partial\xi\partial\eta}\right|_{(1,0)} &= c^7_{1,1} + 2c^7_{2,1} + 3c^7_{3,1} + 4c^7_{4,1} = 0\\
\left.\frac{\partial^2\varphi_7}{\partial\xi\partial\eta}\right|_{(0,1)} &= c^7_{1,1} + 2c^7_{1,2} + 3c^7_{1,2} + 4c^7_{1,4} = 0\\
\left.\frac{\partial^2\varphi_7}{\partial\eta^2}\right|_{(0,0)} &= 2c^0_{0,2} = 0\\
\left.\frac{\partial^2\varphi_7}{\partial\eta^2}\right|_{(1,0)} &= 2c^7_{0,2} + 2c^7_{1,2} + 2c^7_{2,2} + 2c^7_{3,2} = 0\\
\left.\frac{\partial^2\varphi_7}{\partial\eta^2}\right|_{(0,1)} &= 2c^7_{0,2} + 6c^7_{0,3} + 12c^7_{0,4} + 20c^7_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\left(\frac{\partial\varphi_7}{\partial\xi}n_\xi + \frac{\partial\varphi_7}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_7}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^7_{0,1} + \frac{1}{2}c^7_{1,1} + \frac{1}{4}c^7_{2,1} + \frac{1}{8}c^7_{3,1} + \frac{1}{16}c^7_{4,1} = 0\\
\left.\left(\frac{\partial\varphi_7}{\partial\xi}n_\xi + \frac{\partial\varphi_7}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^7_{1,0} + c^7_{2,0} + c^7_{1,1} + \frac{3}{4}c^7_{3,0} + \frac{3}{4}c^7_{2,1} + \frac{3}{4}c^7_{1,2} + \frac{1}{2}c^7_{4,0} +\\
&\quad \frac{1}{2}c^7_{3,1} + \frac{1}{2}c^7_{2,2} + \frac{1}{2}c^7_{1,2} + \frac{5}{16}c^7_{5,0} + \frac{9}{16}c^7_{4,1} + \frac{7}{16}c^7_{3,2} +\\
&\quad \frac{7}{16}c^7_{2,3} + \frac{5}{16}c^7_{1,4} + c^7_{0,1} + c^7_{0,2} + \frac{3}{4}c^7_{0,3} + \frac{1}{2}c^7_{0,4} + \frac{5}{16}c^7_{0,5}) = 0\\
\left.\left(\frac{\partial\varphi_7}{\partial\xi}n_\xi + \frac{\partial\varphi_7}{\partial\eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_7}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^7_{0,1} + c^7_{0,2} + \frac{3}{4}c^7_{0,3} + \frac{1}{2}c^7_{0,4} + \frac{5}{16}c^7_{0,5}\right) = 0
\end{aligned}
$$

The constrains corresponding to φ_8:

$$
\begin{aligned}
\varphi_8|_{(0,0)} &= c^8_{0,0} = 0\\
\varphi_8|_{(1,0)} &= c^8_{0,0} + c^8_{1,0} + c^8_{2,0} + c^8_{3,0} + c^8_{4,0} + c^8_{5,0} = 0\\
\varphi_8|_{(0,1)} &= c^8_{0,0} + c^8_{0,1} + c^8_{0,2} + c^8_{0,3} + c^8_{0,4} + c^8_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_8}{\partial \xi}\right|_{(0,0)} &= c^8_{1,0} = 0 \\
\left.\frac{\partial \varphi_8}{\partial \xi}\right|_{(1,0)} &= c^8_{1,0} + 2c^8_{2,0} + 3c^8_{3,0} + 4c^8_{4,0} + 5c^8_{5,0} = 0 \\
\left.\frac{\partial \varphi_8}{\partial \xi}\right|_{(0,1)} &= c^8_{1,0} + c^8_{1,1}\eta + c^8_{1,2} + c^8_{1,2} + c^8_{1,4} = 0 \\
\left.\frac{\partial \varphi_8}{\partial \eta}\right|_{(0,0)} &= c^8_{0,1} = 0 \\
\left.\frac{\partial \varphi_8}{\partial \eta}\right|_{(1,0)} &= c^8_{0,1} + c^8_{1,1} + c^8_{2,1} + c^8_{3,1} + c^8_{4,1} = 1 \\
\left.\frac{\partial \varphi_8}{\partial \eta}\right|_{(0,1)} &= c^8_{0,1} + 2c^8_{0,2} + 3c^8_{0,3} + 4c^8_{0,4} + 5c^8_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_8}{\partial \xi^2}\right|_{(0,0)} &= 2c^8_{2,0} = 0 \\
\left.\frac{\partial^2 \varphi_8}{\partial \xi^2}\right|_{(1,0)} &= 2c^8_{2,0} + 6c^8_{3,0} + 12c^8_{4,0} + 20c^8_{5,0} = 0 \\
\left.\frac{\partial^2 \varphi_8}{\partial \xi^2}\right|_{(0,1)} &= 2c^8_{2,0} + 2c^8_{2,1} + 2c^8_{2,2} + 2c^8_{2,3} = 0 \\
\left.\frac{\partial^2 \varphi_8}{\partial \xi \partial \eta}\right|_{(0,0)} &= c^8_{1,1} = 0 \\
\left.\frac{\partial^2 \varphi_8}{\partial \xi \partial \eta}\right|_{(1,0)} &= c^8_{1,1} + 2c^8_{2,1} + 3c^8_{3,1} + 4c^8_{4,1} = 0 \\
\left.\frac{\partial^2 \varphi_8}{\partial \xi \partial \eta}\right|_{(0,1)} &= c^8_{1,1} + 2c^8_{1,2} + 3c^8_{1,2} + 4c^8_{1,4} = 0 \\
\left.\frac{\partial^2 \varphi_8}{\partial \eta^2}\right|_{(0,0)} &= 2c^8_{0,2} = 0 \\
\left.\frac{\partial^2 \varphi_8}{\partial \eta^2}\right|_{(1,0)} &= 2c^8_{0,2} + 2c^8_{1,2} + 2c^8_{2,2} + 2c^8_{3,2} = 0 \\
\left.\frac{\partial^2 \varphi_8}{\partial \eta^2}\right|_{(0,1)} &= 2c^8_{0,2} + 6c^{08}_{0,3} + 12c^8_{0,4} + 20c^8_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left(\frac{\partial\varphi_8}{\partial\xi}n_\xi+\frac{\partial\varphi_8}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_8}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^8_{0,1}+\frac{1}{2}c^8_{1,1}+\frac{1}{4}c^8_{2,1}+\frac{1}{8}c^8_{3,1}+\frac{1}{16}c^8_{4,1}=0\\
\left(\frac{\partial\varphi_8}{\partial\xi}n_\xi+\frac{\partial\varphi_8}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^8_{1,0}+c^8_{2,0}+c^8_{1,1}+\frac{3}{4}c^8_{3,0}+\frac{3}{4}c^8_{2,1}+\frac{3}{4}c^8_{1,2}+\frac{1}{2}c^8_{4,0}+\\
&\quad \frac{1}{2}c^8_{3,1}+\frac{1}{2}c^8_{2,2}+\frac{1}{2}c^8_{1,2}+\frac{5}{16}c^8_{5,0}+\frac{9}{16}c^8_{4,1}+\frac{7}{16}c^8_{3,2}+\\
&\quad \frac{7}{16}c^8_{2,3}+\frac{5}{16}c^8_{1,4}+c^8_{0,1}+c^8_{0,2}+\frac{3}{4}c^8_{0,3}+\frac{1}{2}c^8_{0,4}+\frac{5}{16}c^8_{0,5})=0\\
\left(\frac{\partial\varphi_8}{\partial\xi}n_\xi+\frac{\partial\varphi_8}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_8}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^8_{0,1}+c^8_{0,2}+\frac{3}{4}c^8_{0,3}+\frac{1}{2}c^8_{0,4}+\frac{5}{16}c^8_{0,5}\right)=0
\end{aligned}
$$

The constrains corresponding to φ_9:

$$
\begin{aligned}
\varphi_9|_{(0,0)} &= c^9_{0,0}=0\\
\varphi_9|_{(1,0)} &= c^9_{0,0}+c^9_{1,0}+c^9_{2,0}+c^9_{3,0}+c^9_{4,0}+c^9_{5,0}=0\\
\varphi_9|_{(0,1)} &= c^9_{0,0}+c^9_{0,1}+c^9_{0,2}+c^9_{0,3}+c^9_{0,4}+c^9_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial\varphi_9}{\partial\xi}\right|_{(0,0)} &= c^9_{1,0}=0\\
\left.\frac{\partial\varphi_9}{\partial\xi}\right|_{(1,0)} &= c^9_{1,0}+2c^9_{2,0}+3c^9_{3,0}+4c^9_{4,0}+5c^9_{5,0}=0\\
\left.\frac{\partial\varphi_9}{\partial\xi}\right|_{(0,1)} &= c^9_{1,0}+c^9_{1,1}\eta+c^9_{1,2}+c^9_{1,2}+c^9_{1,4}=0\\
\left.\frac{\partial\varphi_9}{\partial\eta}\right|_{(0,0)} &= c^9_{0,1}=0\\
\left.\frac{\partial\varphi_9}{\partial\eta}\right|_{(1,0)} &= c^9_{0,1}+c^9_{1,1}+c^9_{2,1}+c^9_{3,1}+c^9_{4,1}=0\\
\left.\frac{\partial\varphi_9}{\partial\eta}\right|_{(0,1)} &= c^9_{0,1}+2c^9_{0,2}+3c^9_{0,3}+4c^9_{0,4}+5c^9_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2\varphi_9}{\partial\xi^2}\right|_{(0,0)} &= 2c^9_{2,0} = 0\\
\left.\frac{\partial^2\varphi_9}{\partial\xi^2}\right|_{(1,0)} &= 2c^9_{2,0} + 6c^9_{3,0} + 12c^9_{4,0} + 20c^9_{5,0} = 1\\
\left.\frac{\partial^2\varphi_9}{\partial\xi^2}\right|_{(0,1)} &= 2c^9_{2,0} + 2c^9_{2,1} + 2c^9_{2,2} + 2c^9_{2,3} = 0\\
\left.\frac{\partial^2\varphi_9}{\partial\xi\partial\eta}\right|_{(0,0)} &= c^9_{1,1} = 0\\
\left.\frac{\partial^2\varphi_9}{\partial\xi\partial\eta}\right|_{(1,0)} &= c^9_{1,1} + 2c^9_{2,1} + 3c^9_{3,1} + 4c^9_{4,1} = 0\\
\left.\frac{\partial^2\varphi_9}{\partial\xi\partial\eta}\right|_{(0,1)} &= c^9_{1,1} + 2c^9_{1,2} + 3c^9_{1,2} + 4c^9_{1,4} = 0\\
\left.\frac{\partial^2\varphi_9}{\partial\eta^2}\right|_{(0,0)} &= 2c^9_{0,2} = 0\\
\left.\frac{\partial^2\varphi_9}{\partial\eta^2}\right|_{(1,0)} &= 2c^9_{0,2} + 2c^9_{1,2} + 2c^9_{2,2} + 2c^9_{3,2} = 0\\
\left.\frac{\partial^2\varphi_9}{\partial\eta^2}\right|_{(0,1)} &= 2c^9_{0,2} + 6c^9_{0,3} + 12c^9_{0,4} + 20c^9_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\left(\frac{\partial\varphi_9}{\partial\xi}n_\xi + \frac{\partial\varphi_9}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_9}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^9_{0,1} + \frac{1}{2}c^9_{1,1} + \frac{1}{4}c^9_{2,1} + \frac{1}{8}c^9_{3,1} + \frac{1}{16}c^9_{4,1} = 0\\
\left.\left(\frac{\partial\varphi_9}{\partial\xi}n_\xi + \frac{\partial\varphi_9}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^9_{1,0} + c^9_{2,0} + c^9_{1,1} + \frac{3}{4}c^9_{3,0} + \frac{3}{4}c^9_{2,1} + \frac{3}{4}c^9_{1,2} + \frac{1}{2}c^9_{4,0} +\\
&\quad \frac{1}{2}c^9_{3,1} + \frac{1}{2}c^9_{2,2} + \frac{1}{2}c^9_{1,2} + \frac{5}{16}c^9_{5,0} + \frac{9}{16}c^9_{4,1} + \frac{7}{16}c^9_{3,2} +\\
&\quad \frac{7}{16}c^9_{2,3} + \frac{5}{16}c^9_{1,4} + c^9_{0,1} + c^9_{0,2} + \frac{3}{4}c^9_{0,3} + \frac{1}{2}c^9_{0,4} + \frac{5}{16}c^9_{0,5}) = 0\\
\left.\left(\frac{\partial\varphi_9}{\partial\xi}n_\xi + \frac{\partial\varphi_9}{\partial\eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_9}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^9_{0,1} + c^9_{0,2} + \frac{3}{4}c^9_{0,3} + \frac{1}{2}c^9_{0,4} + \frac{5}{16}c^9_{0,5}\right) = 0
\end{aligned}
$$

The constrains corresponding to φ_{10}:

$$
\begin{aligned}
\varphi_{10}|_{(0,0)} &= c^{10}_{0,0} = 0\\
\varphi_{10}|_{(1,0)} &= c^{10}_{0,0} + c^{10}_{1,0} + c^{10}_{2,0} + c^{10}_{3,0} + c^{10}_{4,0} + c^{10}_{5,0} = 0\\
\varphi_{10}|_{(0,1)} &= c^{10}_{0,0} + c^{10}_{0,1} + c^{10}_{0,2} + c^{10}_{0,3} + c^{10}_{0,4} + c^{10}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_{10}}{\partial \xi}\right|_{(0,0)} &= c^{10}_{1,0} = 0 \\
\left.\frac{\partial \varphi 1_{0}}{\partial \xi}\right|_{(1,0)} &= c^{10}_{1,0} + 2c^{10}_{2,0} + 3c^{10}_{3,0} + 4c^{10}_{4,0} + 5c^{10}_{5,0} = 0 \\
\left.\frac{\partial \varphi_{10}}{\partial \xi}\right|_{(0,1)} &= c^{10}_{1,0} + c^{10}_{1,1}\eta + c^{10}_{1,2} + c^{10}_{1,2} + c^{10}_{1,4} = 0 \\
\left.\frac{\partial \varphi_{10}}{\partial \eta}\right|_{(0,0)} &= c^{10}_{0,1} = 0 \\
\left.\frac{\partial \varphi_{10}}{\partial \eta}\right|_{(1,0)} &= c^{10}_{0,1} + c^{10}_{1,1} + c^{10}_{2,1} + c^{10}_{3,1} + c^{10}_{4,1} = 0 \\
\left.\frac{\partial \varphi_{10}}{\partial \eta}\right|_{(0,1)} &= c^{10}_{0,1} + 2c^{10}_{0,2} + 3c^{10}_{0,3} + 4c^{10}_{0,4} + 5c^{10}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_{10}}{\partial \xi^2}\right|_{(0,0)} &= 2c^{10}_{2,0} = 0 \\
\left.\frac{\partial^2 \varphi_{10}}{\partial \xi^2}\right|_{(1,0)} &= 2c^{10}_{2,0} + 6c^{10}_{3,0} + 12c^{10}_{4,0} + 20c^{10}_{5,0} = 0 \\
\left.\frac{\partial^2 \varphi_{10}}{\partial \xi^2}\right|_{(0,1)} &= 2c^{10}_{2,0} + 2c^{10}_{2,1} + 2c^{10}_{2,2} + 2c^{10}_{2,3} = 0 \\
\left.\frac{\partial^2 \varphi_{10}}{\partial \xi \partial \eta}\right|_{(0,0)} &= c^{10}_{1,1} = 0 \\
\left.\frac{\partial^2 \varphi_{10}}{\partial \xi \partial \eta}\right|_{(1,0)} &= c^{10}_{1,1} + 2c^{10}_{2,1} + 3c^{10}_{3,1} + 4c^{10}_{4,1} = 1 \\
\left.\frac{\partial^2 \varphi_{10}}{\partial \xi \partial \eta}\right|_{(0,1)} &= c^{10}_{1,1} + 2c^{10}_{1,2} + 3c^{10}_{1,2} + 4c^{10}_{1,4} = 0 \\
\left.\frac{\partial^2 \varphi_{10}}{\partial \eta^2}\right|_{(0,0)} &= 2c^{10}_{0,2} = 0 \\
\left.\frac{\partial^2 \varphi_{0}}{\partial \eta^2}\right|_{(1,0)} &= 2c^{10}_{0,2} + 2c^{10}_{1,2} + 2c^{10}_{2,2} + 2c^{10}_{3,2} = 0 \\
\left.\frac{\partial^2 \varphi_{0}}{\partial \eta^2}\right|_{(0,1)} &= 2c^{10}_{0,2} + 6c^{10}_{0,3} + 12c^{10}_{0,4} + 20c^{10}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left(\frac{\partial\varphi_{10}}{\partial\xi}n_\xi+\frac{\partial\varphi_{10}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} &= \frac{\partial\varphi_{10}}{\partial\eta}\Big|_{\left(\frac{1}{2},0\right)} = c^{10}_{0,1}+\frac{1}{2}c^{10}_{1,1}+\frac{1}{4}c^{10}_{2,1}+\frac{1}{8}c^{10}_{3,1}+\frac{1}{16}c^{10}_{4,1}=0\\
\left(\frac{\partial\varphi_{10}}{\partial\xi}n_\xi+\frac{\partial\varphi_{10}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{10}_{1,0}+c^{10}_{2,0}+c^{10}_{1,1}+\frac{3}{4}c^{10}_{3,0}+\frac{3}{4}c^{10}_{2,1}+\frac{3}{4}c^{10}_{1,2}+\frac{1}{2}c^{10}_{4,0}+\\
&\quad \frac{1}{2}c^{10}_{3,1}+\frac{1}{2}c^{10}_{2,2}+\frac{1}{2}c^{10}_{1,2}+\frac{5}{16}c^{10}_{5,0}+\frac{9}{16}c^{10}_{4,1}+\frac{7}{16}c^{10}_{3,2}+\\
&\quad \frac{7}{16}c^{10}_{2,3}+\frac{5}{16}c^{10}_{1,4}+c^{10}_{0,1}+c^{10}_{0,2}+\frac{3}{4}c^{10}_{0,3}+\frac{1}{2}c^{10}_{0,4}+\frac{5}{16}c^{10}_{0,5})=0\\
\left(\frac{\partial\varphi_{10}}{\partial\xi}n_\xi+\frac{\partial\varphi_{10}}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} &= -\frac{\partial\varphi_{10}}{\partial\xi}\Big|_{\left(0,\frac{1}{2}\right)} = -\left(c^{10}_{0,1}+c^{10}_{0,2}+\frac{3}{4}c^{10}_{0,3}+\frac{1}{2}c^{10}_{0,4}+\frac{5}{16}c^{10}_{0,5}\right)=0
\end{aligned}
$$

The constrains corresponding to φ_{11}:

$$
\begin{aligned}
\varphi_{11}|_{(0,0)} &= c^{11}_{0,0}=0\\
\varphi_{11}|_{(1,0)} &= c^{11}_{0,0}+c^{11}_{1,0}+c^{11}_{2,0}+c^{11}_{3,0}+c^{11}_{4,0}+c^{11}_{5,0}=0\\
\varphi_{11}|_{(0,1)} &= c^{11}_{0,0}+c^{11}_{0,1}+c^{11}_{0,2}+c^{11}_{0,3}+c^{11}_{0,4}+c^{11}_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial\varphi_{11}}{\partial\xi}\Big|_{(0,0)} &= c^{11}_{1,0}=0\\
\frac{\partial\varphi_{11}}{\partial\xi}\Big|_{(1,0)} &= c^{11}_{1,0}+2c^{11}_{2,0}+3c^{11}_{3,0}+4c^{11}_{4,0}+5c^{11}_{5,0}=0\\
\frac{\partial\varphi_{11}}{\partial\xi}\Big|_{(0,1)} &= c^{11}_{1,0}+c^{11}_{1,1}\eta+c^{11}_{1,2}+c^{11}_{1,2}+c^{11}_{1,4}=0\\
\frac{\partial\varphi_{11}}{\partial\eta}\Big|_{(0,0)} &= c^{11}_{0,1}=0\\
\frac{\partial\varphi_{11}}{\partial\eta}\Big|_{(1,0)} &= c^{11}_{0,1}+c^{11}_{1,1}+c^{11}_{2,1}+c^{11}_{3,1}+c^{11}_{4,1}=0\\
\frac{\partial\varphi_{11}}{\partial\eta}\Big|_{(0,1)} &= c^{11}_{0,1}+2c^{11}_{0,2}+3c^{11}_{0,3}+4c^{11}_{0,4}+5c^{11}_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2\varphi_{11}}{\partial\xi^2}\right|_{(0,0)} &= 2c^{11}_{2,0} = 0\\
\left.\frac{\partial^2\varphi_{11}}{\partial\xi^2}\right|_{(1,0)} &= 2c^{11}_{2,0} + 6c^{11}_{3,0} + 12c^{11}_{4,0} + 20c^{11}_{5,0} = 0\\
\left.\frac{\partial^2\varphi_{11}}{\partial\xi^2}\right|_{(0,1)} &= 2c^{11}_{2,0} + 2c^{11}_{2,1} + 2c^{11}_{2,2} + 2c^{11}_{2,3} = 0\\
\left.\frac{\partial^2\varphi_{11}}{\partial\xi\partial\eta}\right|_{(0,0)} &= c^{11}_{1,1} = 0\\
\left.\frac{\partial^2\varphi_{11}}{\partial\xi\partial\eta}\right|_{(1,0)} &= c^{11}_{1,1} + 2c^{11}_{2,1} + 3c^{11}_{3,1} + 4c^{11}_{4,1} = 0\\
\left.\frac{\partial^2\varphi_{11}}{\partial\xi\partial\eta}\right|_{(0,1)} &= c^{11}_{1,1} + 2c^{11}_{1,2} + 3c^{11}_{1,2} + 4c^{11}_{1,4} = 0\\
\left.\frac{\partial^2\varphi_{11}}{\partial\eta^2}\right|_{(0,0)} &= 2c^{11}_{0,2} = 0\\
\left.\frac{\partial^2\varphi_{11}}{\partial\eta^2}\right|_{(1,0)} &= 2c^{11}_{0,2} + 2c^{11}_{1,2} + 2c^{11}_{2,2} + 2c^{11}_{3,2} = 1\\
\left.\frac{\partial^2\varphi_{11}}{\partial\eta^2}\right|_{(0,1)} &= 2c^{11}_{0,2} + 6c^{11}_{0,3} + 12c^{11}_{0,4} + 20c^{11}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\left(\frac{\partial\varphi_{11}}{\partial\xi}n_\xi + \frac{\partial\varphi_{11}}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_{11}}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^{11}_{0,1} + \frac{1}{2}c^{11}_{1,1} + \frac{1}{4}c^{11}_{2,1} + \frac{1}{8}c^{11}_{3,1} + \frac{1}{16}c^{11}_{4,1} = 0\\
\left.\left(\frac{\partial\varphi_{11}}{\partial\xi}n_\xi + \frac{\partial\varphi_{11}}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{11}_{1,0} + c^{11}_{2,0} + c^{11}_{1,1} + \frac{3}{4}c^{11}_{3,0} + \frac{3}{4}c^{11}_{2,1} + \frac{3}{4}c^{11}_{1,2} + \frac{1}{2}c^{11}_{4,0} +\\
&\quad \frac{1}{2}c^{11}_{3,1} + \frac{1}{2}c^{11}_{2,2} + \frac{1}{2}c^{11}_{1,2} + \frac{5}{16}c^{11}_{5,0} + \frac{9}{16}c^{11}_{4,1} + \frac{7}{16}c^{11}_{3,2} +\\
&\quad \frac{7}{16}c^{11}_{2,3} + \frac{5}{16}c^{11}_{1,4} + c^{11}_{0,1} + c^{11}_{0,2} + \frac{3}{4}c^{11}_{0,3} + \frac{1}{2}c^{11}_{0,4} + \frac{5}{16}c^{11}_{0,5}) = 0\\
\left.\left(\frac{\partial\varphi_{11}}{\partial\xi}n_\xi + \frac{\partial\varphi_{11}}{\partial\eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_{11}}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^{11}_{0,1} + c^{11}_{0,2} + \frac{3}{4}c^{11}_{0,3} + \frac{1}{2}c^{11}_{0,4} + \frac{5}{16}c^{11}_{0,5}\right) = 0
\end{aligned}
$$

The constrains corresponding to φ_{12}:

$$
\begin{aligned}
\varphi_{12}|_{(0,0)} &= c^{12}_{0,0} = 0\\
\varphi_{12}|_{(1,0)} &= c^{12}_{0,0} + c^{12}_{1,0} + c^{12}_{2,0} + c^{12}_{3,0} + c^{12}_{4,0} + c^{12}_{5,0} = 0\\
\varphi_{12}|_{(0,1)} &= c^{12}_{0,0} + c^{12}_{0,1} + c^{12}_{0,2} + c^{12}_{0,3} + c^{12}_{0,4} + c^{12}_{0,5} = 1
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_{12}}{\partial \xi}\right|_{(0,0)} &= c_{1,0}^{12} = 0 \\
\left.\frac{\partial \varphi_{12}}{\partial \xi}\right|_{(1,0)} &= c_{1,0}^{12} + 2c_{2,0}^{12} + 3c_{3,0}^{12} + 4c_{4,0}^{12} + 5c_{5,0}^{12} = 0 \\
\left.\frac{\partial \varphi_{12}}{\partial \xi}\right|_{(0,1)} &= c_{1,0}^{12} + c_{1,1}^{12}\eta + c_{1,2}^{12} + c_{1,2}^{12} + c_{1,4}^{12} = 0 \\
\left.\frac{\partial \varphi_{12}}{\partial \eta}\right|_{(0,0)} &= c_{0,1}^{12} = 0 \\
\left.\frac{\partial \varphi_{12}}{\partial \eta}\right|_{(1,0)} &= c_{0,1}^{12} + c_{1,1}^{12} + c_{2,1}^{12} + c_{3,1}^{12} + c_{4,1}^{12} = 0 \\
\left.\frac{\partial \varphi_{12}}{\partial \eta}\right|_{(0,1)} &= c_{0,1}^{12} + 2c_{0,2}^{12} + 3c_{0,3}^{12} + 4c_{0,4}^{12} + 5c_{0,5}^{12} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_{12}}{\partial \xi^2}\right|_{(0,0)} &= 2c_{2,0}^{12} = 0 \\
\left.\frac{\partial^2 \varphi_{12}}{\partial \xi^2}\right|_{(1,0)} &= 2c_{2,0}^{12} + 6c_{3,0}^{12} + 12c_{4,0}^{12} + 20c_{5,0}^{12} = 0 \\
\left.\frac{\partial^2 \varphi_{12}}{\partial \xi^2}\right|_{(0,1)} &= 2c_{2,0}^{12} + 2c_{2,1}^{12} + 2c_{2,2}^{12} + 2c_{2,3}^{12} = 0 \\
\left.\frac{\partial^2 \varphi_{12}}{\partial \xi \partial \eta}\right|_{(0,0)} &= c_{1,1}^{12} = 0 \\
\left.\frac{\partial^2 \varphi_{12}}{\partial \xi \partial \eta}\right|_{(1,0)} &= c_{1,1}^{12} + 2c_{2,1}^{12} + 3c_{3,1}^{12} + 4c_{4,1}^{12} = 0 \\
\left.\frac{\partial^2 \varphi_{12}}{\partial \xi \partial \eta}\right|_{(0,1)} &= c_{1,1}^{12} + 2c_{1,2}^{12} + 3c_{1,2}^{12} + 4c_{1,4}^{12} = 0 \\
\left.\frac{\partial^2 \varphi_{12}}{\partial \eta^2}\right|_{(0,0)} &= 2c_{0,2}^{12} = 0 \\
\left.\frac{\partial^2 \varphi_{12}}{\partial \eta^2}\right|_{(1,0)} &= 2c_{0,2}^{12} + 2c_{1,2}^{12} + 2c_{2,2}^{12} + 2c_{3,2}^{12} = 0 \\
\left.\frac{\partial^2 \varphi_{12}}{\partial \eta^2}\right|_{(0,1)} &= 2c_{0,2}^{12} + 6c_{0,3}^{12} + 12c_{0,4}^{12} + 20c_{0,5}^{12} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left(\frac{\partial\varphi_{12}}{\partial\xi}n_\xi+\frac{\partial\varphi_{12}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} &= \frac{\partial\varphi_{12}}{\partial\eta}\Big|_{\left(\frac{1}{2},0\right)} = c^{12}_{0,1}+\frac{1}{2}c^{12}_{1,1}+\frac{1}{4}c^{12}_{2,1}+\frac{1}{8}c^{12}_{3,1}+\frac{1}{16}c^{12}_{4,1}=0\\
\left(\frac{\partial\varphi_{12}}{\partial\xi}n_\xi+\frac{\partial\varphi_{12}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{12}_{1,0}+c^{12}_{2,0}+c^{12}_{1,1}+\frac{3}{4}c^{12}_{3,0}+\frac{3}{4}c^{12}_{2,1}+\frac{3}{4}c^{12}_{1,2}+\frac{1}{2}c^{12}_{4,0}+\\
&\quad \frac{1}{2}c^{12}_{3,1}+\frac{1}{2}c^{12}_{2,2}+\frac{1}{2}c^{12}_{1,2}+\frac{5}{16}c^{12}_{5,0}+\frac{9}{16}c^{12}_{4,1}+\frac{7}{16}c^{12}_{3,2}+\\
&\quad \frac{7}{16}c^{12}_{2,3}+\frac{5}{16}c^{12}_{1,4}+c^{12}_{0,1}+c^{12}_{0,2}+\frac{3}{4}c^{12}_{0,3}+\frac{1}{2}c^{12}_{0,4}+\frac{5}{16}c^{12}_{0,5})=0\\
\left(\frac{\partial\varphi_{12}}{\partial\xi}n_\xi+\frac{\partial\varphi_{12}}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} &= -\frac{\partial\varphi_{12}}{\partial\xi}\Big|_{\left(0,\frac{1}{2}\right)} = -\left(c^{12}_{0,1}+c^{12}_{0,2}+\frac{3}{4}c^{12}_{0,3}+\frac{1}{2}c^{12}_{0,4}+\frac{5}{16}c^{12}_{0,5}\right)=0
\end{aligned}
$$

The constrains corresponding to φ_{13}:

$$
\begin{aligned}
\varphi_{13}|_{(0,0)} &= c^{13}_{0,0}=0\\
\varphi_{13}|_{(1,0)} &= c^{13}_{0,0}+c^{13}_{1,0}+c^{13}_{2,0}+c^{13}_{3,0}+c^{13}_{4,0}+c^{13}_{5,0}=0\\
\varphi_{13}|_{(0,1)} &= c^{13}_{0,0}+c^{13}_{0,1}+c^{13}_{0,2}+c^{13}_{0,3}+c^{13}_{0,4}+c^{13}_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial\varphi_{13}}{\partial\xi}\Big|_{(0,0)} &= c^{13}_{1,0}=0\\
\frac{\partial\varphi_{13}}{\partial\xi}\Big|_{(1,0)} &= c^{13}_{1,0}+2c^{13}_{2,0}+3c^{13}_{3,0}+4c^{13}_{4,0}+5c^{13}_{5,0}=0\\
\frac{\partial\varphi_{13}}{\partial\xi}\Big|_{(0,1)} &= c^{13}_{1,0}+c^{13}_{1,1}\eta+c^{13}_{1,2}+c^{13}_{1,2}+c^{13}_{1,4}=1\\
\frac{\partial\varphi_{13}}{\partial\eta}\Big|_{(0,0)} &= c^{13}_{0,1}=0\\
\frac{\partial\varphi_{13}}{\partial\eta}\Big|_{(1,0)} &= c^{13}_{0,1}+c^{13}_{1,1}+c^{13}_{2,1}+c^{13}_{3,1}+c^{13}_{4,1}=0\\
\frac{\partial\varphi_{13}}{\partial\eta}\Big|_{(0,1)} &= c^{13}_{0,1}+2c^{13}_{0,2}+3c^{13}_{0,3}+4c^{13}_{0,4}+5c^{13}_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2\varphi_{13}}{\partial\xi^2}\right|_{(0,0)} &= 2c^{13}_{2,0} = 0\\
\left.\frac{\partial^2\varphi_{13}}{\partial\xi^2}\right|_{(1,0)} &= 2c^{13}_{2,0} + 6c^{13}_{3,0} + 12c^{13}_{4,0} + 20c^{13}_{5,0} = 0\\
\left.\frac{\partial^2\varphi_{13}}{\partial\xi^2}\right|_{(0,1)} &= 2c^{13}_{2,0} + 2c^{13}_{2,1} + 2c^{13}_{2,2} + 2c^{13}_{2,3} = 0\\
\left.\frac{\partial^2\varphi_{13}}{\partial\xi\partial\eta}\right|_{(0,0)} &= c^{13}_{1,1} = 0\\
\left.\frac{\partial^2\varphi_{13}}{\partial\xi\partial\eta}\right|_{(1,0)} &= c^{13}_{1,1} + 2c^{13}_{2,1} + 3c^{13}_{3,1} + 4c^{13}_{4,1} = 0\\
\left.\frac{\partial^2\varphi_{13}}{\partial\xi\partial\eta}\right|_{(0,1)} &= c^{13}_{1,1} + 2c^{13}_{1,2} + 3c^{13}_{1,2} + 4c^{13}_{1,4} = 0\\
\left.\frac{\partial^2\varphi_{13}}{\partial\eta^2}\right|_{(0,0)} &= 2c^{13}_{0,2} = 0\\
\left.\frac{\partial^2\varphi_{13}}{\partial\eta^2}\right|_{(1,0)} &= 2c^{13}_{0,2} + 2c^{13}_{1,2} + 2c^{13}_{2,2} + 2c^{13}_{3,2} = 0\\
\left.\frac{\partial^2\varphi_{13}}{\partial\eta^2}\right|_{(0,1)} &= 2c^{13}_{0,2} + 6c^{13}_{0,3} + 12c^{13}_{0,4} + 20c^{13}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\left(\frac{\partial\varphi_{13}}{\partial\xi}n_\xi + \frac{\partial\varphi_{13}}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_{13}}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^{13}_{0,1} + \frac{1}{2}c^{13}_{1,1} + \frac{1}{4}c^{13}_{2,1} + \frac{1}{8}c^{13}_{3,1} + \frac{1}{16}c^{13}_{4,1} = 0\\
\left.\left(\frac{\partial\varphi_{13}}{\partial\xi}n_\xi + \frac{\partial\varphi_{13}}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{13}_{1,0} + c^{13}_{2,0} + c^{13}_{1,1} + \frac{3}{4}c^{13}_{3,0} + \frac{3}{4}c^{13}_{2,1} + \frac{3}{4}c^{13}_{1,2} + \frac{1}{2}c^{13}_{4,0} +\\
&\quad \frac{1}{2}c^{13}_{3,1} + \frac{1}{2}c^{13}_{2,2} + \frac{1}{2}c^{13}_{1,2} + \frac{5}{16}c^{13}_{5,0} + \frac{9}{16}c^{13}_{4,1} + \frac{7}{16}c^{13}_{3,2} +\\
&\quad \frac{7}{16}c^{13}_{2,3} + \frac{5}{16}c^{13}_{1,4} + c^{13}_{0,1} + c^{13}_{0,2} + \frac{3}{4}c^{13}_{0,3} + \frac{1}{2}c^{13}_{0,4} + \frac{5}{16}c^{13}_{0,5}) = 0\\
\left.\left(\frac{\partial\varphi_{13}}{\partial\xi}n_\xi + \frac{\partial\varphi_{13}}{\partial\eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_{13}}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^{13}_{0,1} + c^{13}_{0,2} + \frac{3}{4}c^{13}_{0,3} + \frac{1}{2}c^{13}_{0,4} + \frac{5}{16}c^{13}_{0,5}\right) = 0
\end{aligned}
$$

The constrains corresponding to φ_{14}:

$$
\begin{aligned}
\varphi_{14}|_{(0,0)} &= c^{14}_{0,0} = 0\\
\varphi_{14}|_{(1,0)} &= c^{14}_{0,0} + c^{14}_{1,0} + c^{14}_{2,0} + c^{14}_{3,0} + c^{14}_{4,0} + c^{14}_{5,0} = 0\\
\varphi_{14}|_{(0,1)} &= c^{14}_{0,0} + c^{14}_{0,1} + c^{14}_{0,2} + c^{14}_{0,3} + c^{14}_{0,4} + c^{14}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_{14}}{\partial \xi}\right|_{(0,0)} &= c_{1,0}^{14} = 0 \\
\left.\frac{\partial \varphi_{14}}{\partial \xi}\right|_{(1,0)} &= c_{1,0}^{14} + 2c_{2,0}^{14} + 3c_{3,0}^{14} + 4c_{4,0}^{14} + 5c_{5,0}^{14} = 0 \\
\left.\frac{\partial \varphi_{14}}{\partial \xi}\right|_{(0,1)} &= c_{1,0}^{14} + c_{1,1}^{14}\eta + c_{1,2}^{14} + c_{1,2}^{14} + c_{1,4}^{14} = 0 \\
\left.\frac{\partial \varphi_{14}}{\partial \eta}\right|_{(0,0)} &= c_{0,1}^{14} = 0 \\
\left.\frac{\partial \varphi_{14}}{\partial \eta}\right|_{(1,0)} &= c_{0,1}^{14} + c_{1,1}^{14} + c_{2,1}^{14} + c_{3,1}^{14} + c_{4,1}^{14} = 0 \\
\left.\frac{\partial \varphi_{14}}{\partial \eta}\right|_{(0,1)} &= c_{0,1}^{4} + 2c_{0,2}^{14} + 3c_{0,3}^{14} + 4c_{0,4}^{14} + 5c_{0,5}^{14} = 1
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_{14}}{\partial \xi^2}\right|_{(0,0)} &= 2c_{2,0}^{14} = 0 \\
\left.\frac{\partial^2 \varphi_{14}}{\partial \xi^2}\right|_{(1,0)} &= 2c_{2,0}^{14} + 6c_{3,0}^{14} + 12c_{4,0}^{14} + 20c_{5,0}^{14} = 0 \\
\left.\frac{\partial^2 \varphi_{14}}{\partial \xi^2}\right|_{(0,1)} &= 2c_{2,0}^{14} + 2c_{2,1}^{14} + 2c_{2,2}^{14} + 2c_{2,3}^{14} = 0 \\
\left.\frac{\partial^2 \varphi_{14}}{\partial \xi \partial \eta}\right|_{(0,0)} &= c_{1,1}^{14} = 0 \\
\left.\frac{\partial^2 \varphi_{14}}{\partial \xi \partial \eta}\right|_{(1,0)} &= c_{1,1}^{14} + 2c_{2,1}^{14} + 3c_{3,1}^{14} + 4c_{4,1}^{14} = 0 \\
\left.\frac{\partial^2 \varphi_{14}}{\partial \xi \partial \eta}\right|_{(0,1)} &= c_{1,1}^{14} + 2c_{1,2}^{14} + 3c_{1,2}^{14} + 4c_{1,4}^{14} = 0 \\
\left.\frac{\partial^2 \varphi_{14}}{\partial \eta^2}\right|_{(0,0)} &= 2c_{0,2}^{14} = 0 \\
\left.\frac{\partial^2 \varphi_{14}}{\partial \eta^2}\right|_{(1,0)} &= 2c_{0,2}^{14} + 2c_{1,2}^{14} + 2c_{2,2}^{14} + 2c_{3,2}^{14} = 0 \\
\left.\frac{\partial^2 \varphi_{14}}{\partial \eta^2}\right|_{(0,1)} &= 2c_{0,2}^{14} + 6c_{0,3}^{14} + 12c_{0,4}^{14} + 20c_{0,5}^{14} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left(\frac{\partial\varphi_{14}}{\partial\xi}n_\xi+\frac{\partial\varphi_{14}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} &= \frac{\partial\varphi_{14}}{\partial\eta}\Big|_{\left(\frac{1}{2},0\right)} = c^{14}_{0,1}+\frac{1}{2}c^{14}_{1,1}+\frac{1}{4}c^{14}_{2,1}+\frac{1}{8}c^{14}_{3,1}+\frac{1}{16}c^{14}_{4,1}=0\\
\left(\frac{\partial\varphi_{14}}{\partial\xi}n_\xi+\frac{\partial\varphi_{14}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{14}_{1,0}+c^{14}_{2,0}+c^{14}_{1,1}+\frac{3}{4}c^{14}_{3,0}+\frac{3}{4}c^{14}_{2,1}+\frac{3}{4}c^{14}_{1,2}+\frac{1}{2}c^{14}_{4,0}+\\
&\quad \frac{1}{2}c^{14}_{3,1}+\frac{1}{2}c^{14}_{2,2}+\frac{1}{2}c^{14}_{1,2}+\frac{5}{16}c^{14}_{5,0}+\frac{9}{16}c^{14}_{4,1}+\frac{7}{16}c^{14}_{3,2}+\\
&\quad \frac{7}{16}c^{14}_{2,3}+\frac{5}{16}c^{14}_{1,4}+c^{14}_{0,1}+c^{14}_{0,2}+\frac{3}{4}c^{14}_{0,3}+\frac{1}{2}c^{14}_{0,4}+\frac{5}{16}c^{14}_{0,5})=0\\
\left(\frac{\partial\varphi_{14}}{\partial\xi}n_\xi+\frac{\partial\varphi_{14}}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} &= -\frac{\partial\varphi_{14}}{\partial\xi}\Big|_{\left(0,\frac{1}{2}\right)} = -\left(c^{14}_{0,1}+c^{14}_{0,2}+\frac{3}{4}c^{14}_{0,3}+\frac{1}{2}c^{14}_{0,4}+\frac{5}{16}c^{14}_{0,5}\right)=0
\end{aligned}
$$

The constrains corresponding to φ_{15}:

$$
\begin{aligned}
\varphi_{15}|_{(0,0)} &= c^{15}_{0,0}=0\\
\varphi_{15}|_{(1,0)} &= c^{15}_{0,0}+c^{15}_{1,0}+c^{15}_{2,0}+c^{15}_{3,0}+c^{15}_{4,0}+c^{15}_{5,0}=0\\
\varphi_{15}|_{(0,1)} &= c^{15}_{0,0}+c^{15}_{0,1}+c^{15}_{0,2}+c^{15}_{0,3}+c^{5}_{0,4}+c^{15}_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial\varphi_{15}}{\partial\xi}\Big|_{(0,0)} &= c^{15}_{1,0}=0\\
\frac{\partial\varphi_{15}}{\partial\xi}\Big|_{(1,0)} &= c^{15}_{1,0}+2c^{15}_{2,0}+3c^{15}_{3,0}+4c^{15}_{4,0}+5c^{15}_{5,0}=0\\
\frac{\partial\varphi_{15}}{\partial\xi}\Big|_{(0,1)} &= c^{15}_{1,0}+c^{15}_{1,1}\eta+c^{15}_{1,2}+c^{15}_{1,2}+c^{15}_{1,4}=0\\
\frac{\partial\varphi_{15}}{\partial\eta}\Big|_{(0,0)} &= c^{15}_{0,1}=0\\
\frac{\partial\varphi_{15}}{\partial\eta}\Big|_{(1,0)} &= c^{15}_{0,1}+c^{15}_{1,1}+c^{15}_{2,1}+c^{15}_{3,1}+c^{15}_{4,1}=0\\
\frac{\partial\varphi_{15}}{\partial\eta}\Big|_{(0,1)} &= c^{15}_{0,1}+2c^{15}_{0,2}+3c^{15}_{0,3}+4c^{15}_{0,4}+5c^{15}_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2\varphi_{15}}{\partial\xi^2}\right|_{(0,0)} &= 2c^{15}_{2,0} = 0\\
\left.\frac{\partial^2\varphi_{15}}{\partial\xi^2}\right|_{(1,0)} &= 2c^{15}_{2,0} + 6c^{15}_{3,0} + 12c^{15}_{4,0} + 20c^{15}_{5,0} = 0\\
\left.\frac{\partial^2\varphi_{15}}{\partial\xi^2}\right|_{(0,1)} &= 2c^{15}_{2,0} + 2c^{15}_{2,1} + 2c^{15}_{2,2} + 2c^{15}_{2,3} = 1\\
\left.\frac{\partial^2\varphi_{15}}{\partial\xi\partial\eta}\right|_{(0,0)} &= c^{15}_{1,1} = 0\\
\left.\frac{\partial^2\varphi_{15}}{\partial\xi\partial\eta}\right|_{(1,0)} &= c^{15}_{1,1} + 2c^{15}_{2,1} + 3c^{15}_{3,1} + 4c^{15}_{4,1} = 0\\
\left.\frac{\partial^2\varphi_{15}}{\partial\xi\partial\eta}\right|_{(0,1)} &= c^{15}_{1,1} + 2c^{15}_{1,2} + 3c^{15}_{1,2} + 4c^{15}_{1,4} = 0\\
\left.\frac{\partial^2\varphi_{15}}{\partial\eta^2}\right|_{(0,0)} &= 2c^{15}_{0,2} = 0\\
\left.\frac{\partial^2\varphi_{15}}{\partial\eta^2}\right|_{(1,0)} &= 2c^{15}_{0,2} + 2c^{15}_{1,2} + 2c^{15}_{2,2} + 2c^{15}_{3,2} = 0\\
\left.\frac{\partial^2\varphi_{15}}{\partial\eta^2}\right|_{(0,1)} &= 2c^{15}_{0,2} + 6c^{15}_{0,3} + 12c^{15}_{0,4} + 20c^{15}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\left(\frac{\partial\varphi_{15}}{\partial\xi}n_\xi + \frac{\partial\varphi_{15}}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_{15}}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^{15}_{0,1} + \frac{1}{2}c^{15}_{1,1} + \frac{1}{4}c^{15}_{2,1} + \frac{1}{8}c^{15}_{3,1} + \frac{1}{16}c^{15}_{4,1} = 0\\
\left.\left(\frac{\partial\varphi_{15}}{\partial\xi}n_\xi + \frac{\partial\varphi_{15}}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{15}_{1,0} + c^{15}_{2,0} + c^{15}_{1,1} + \frac{3}{4}c^{15}_{3,0} + \frac{3}{4}c^{15}_{2,1} + \frac{3}{4}c^{15}_{1,2} + \frac{1}{2}c^{15}_{4,0} +\\
&\quad \frac{1}{2}c^{15}_{3,1} + \frac{1}{2}c^{15}_{2,2} + \frac{1}{2}c^{15}_{1,2} + \frac{5}{16}c^{15}_{5,0} + \frac{9}{16}c^{15}_{4,1} + \frac{7}{16}c^{15}_{3,2} +\\
&\quad \frac{7}{16}c^{15}_{2,3} + \frac{5}{16}c^{15}_{1,4} + c^{15}_{0,1} + c^{15}_{0,2} + \frac{3}{4}c^{15}_{0,3} + \frac{1}{2}c^{15}_{0,4} + \frac{5}{16}c^{15}_{0,5}) = 0\\
\left.\left(\frac{\partial\varphi_{15}}{\partial\xi}n_\xi + \frac{\partial\varphi_{15}}{\partial\eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_{15}}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^{15}_{0,1} + c^{15}_{0,2} + \frac{3}{4}c^{15}_{0,3} + \frac{1}{2}c^{15}_{0,4} + \frac{5}{16}c^{15}_{0,5}\right) = 0
\end{aligned}
$$

The constrains corresponding to φ_{16}:

$$
\begin{aligned}
\varphi_{16}|_{(0,0)} &= c^{16}_{0,0} = 0\\
\varphi_{16}|_{(1,0)} &= c^{16}_{0,0} + c^{16}_{1,0} + c^{16}_{2,0} + c^{16}_{3,0} + c^{16}_{4,0} + c^{16}_{5,0} = 0\\
\varphi_{16}|_{(0,1)} &= c^{16}_{0,0} + c^{16}_{0,1} + c^{16}_{0,2} + c^{16}_{0,3} + c^{16}_{0,4} + c^{16}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_{16}}{\partial \xi}\right|_{(0,0)} &= c_{1,0}^{16} = 0 \\
\left.\frac{\partial \varphi_{16}}{\partial \xi}\right|_{(1,0)} &= c_{1,0}^{16} + 2c_{2,0}^{16} + 3c_{3,0}^{16} + 4c_{4,0}^{16} + 5c_{5,0}^{16} = 0 \\
\left.\frac{\partial \varphi_{16}}{\partial \xi}\right|_{(0,1)} &= c_{1,0}^{16} + c_{1,1}^{16}\eta + c_{1,2}^{16} + c_{1,2}^{16} + c_{1,4}^{16} = 0 \\
\left.\frac{\partial \varphi_{16}}{\partial \eta}\right|_{(0,0)} &= c_{0,1}^{16} = 0 \\
\left.\frac{\partial \varphi_{16}}{\partial \eta}\right|_{(1,0)} &= c_{0,1}^{16} + c_{1,1}^{16} + c_{2,1}^{16} + c_{3,1}^{16} + c_{4,1}^{16} = 0 \\
\left.\frac{\partial \varphi_{16}}{\partial \eta}\right|_{(0,1)} &= c_{0,1}^{16} + 2c_{0,2}^{16} + 3c_{0,3}^{16} + 4c_{0,4}^{16} + 5c_{0,5}^{16} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_{16}}{\partial \xi^2}\right|_{(0,0)} &= 2c_{2,0}^{16} = 0 \\
\left.\frac{\partial^2 \varphi_{16}}{\partial \xi^2}\right|_{(1,0)} &= 2c_{2,0}^{16} + 6c_{3,0}^{16} + 12c_{4,0}^{16} + 20c_{5,0}^{16} = 0 \\
\left.\frac{\partial^2 \varphi_{16}}{\partial \xi^2}\right|_{(0,1)} &= 2c_{2,0}^{16} + 2c_{2,1}^{16} + 2c_{2,2}^{16} + 2c_{2,3}^{16} = 0 \\
\left.\frac{\partial^2 \varphi_{16}}{\partial \xi \partial \eta}\right|_{(0,0)} &= c_{1,1}^{16} = 0 \\
\left.\frac{\partial^2 \varphi_{06}}{\partial \xi \partial \eta}\right|_{(1,0)} &= c_{1,1}^{16} + 2c_{2,1}^{16} + 3c_{3,1}^{16} + 4c_{4,1}^{16} = 0 \\
\left.\frac{\partial^2 \varphi_{16}}{\partial \xi \partial \eta}\right|_{(0,1)} &= c_{1,1}^{16} + 2c_{1,2}^{16} + 3c_{1,2}^{16} + 4c_{1,4}^{16} = 1 \\
\left.\frac{\partial^2 \varphi_{16}}{\partial \eta^2}\right|_{(0,0)} &= 2c_{0,2}^{16} = 0 \\
\left.\frac{\partial^2 \varphi_{16}}{\partial \eta^2}\right|_{(1,0)} &= 2c_{0,2}^{16} + 2c_{1,2}^{16} + 2c_{2,2}^{16} + 2c_{3,2}^{16} = 0 \\
\left.\frac{\partial^2 \varphi_{16}}{\partial \eta^2}\right|_{(0,1)} &= 2c_{0,2}^{16} + 6c_{0,3}^{16} + 12c_{0,4}^{16} + 20c_{0,5}^{16} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left(\frac{\partial\varphi_{16}}{\partial\xi}n_\xi+\frac{\partial\varphi_{16}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} &= \frac{\partial\varphi_{16}}{\partial\eta}\Big|_{\left(\frac{1}{2},0\right)} = c^{16}_{0,1}+\frac{1}{2}c^{16}_{1,1}+\frac{1}{4}c^{16}_{2,1}+\frac{1}{8}c^{16}_{3,1}+\frac{1}{16}c^{16}_{4,1}=0\\
\left(\frac{\partial\varphi_{16}}{\partial\xi}n_\xi+\frac{\partial\varphi_{16}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{16}_{1,0}+c^{16}_{2,0}+c^{16}_{1,1}+\frac{3}{4}c^{16}_{3,0}+\frac{3}{4}c^{16}_{2,1}+\frac{3}{4}c^{16}_{1,2}+\frac{1}{2}c^{16}_{4,0}+\\
&\quad \frac{1}{2}c^{16}_{3,1}+\frac{1}{2}c^{16}_{2,2}+\frac{1}{2}c^{16}_{1,2}+\frac{5}{16}c^{16}_{5,0}+\frac{9}{16}c^{16}_{4,1}+\frac{7}{16}c^{16}_{3,2}+\\
&\quad \frac{7}{16}c^{16}_{2,3}+\frac{5}{16}c^{16}_{1,4}+c^{16}_{0,1}+c^{16}_{0,2}+\frac{3}{4}c^{16}_{0,3}+\frac{1}{2}c^{16}_{0,4}+\frac{5}{16}c^{16}_{0,5})=0\\
\left(\frac{\partial\varphi_{16}}{\partial\xi}n_\xi+\frac{\partial\varphi_{16}}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} &= -\frac{\partial\varphi_{16}}{\partial\xi}\Big|_{\left(0,\frac{1}{2}\right)} = -\left(c^{16}_{0,1}+c^{16}_{0,2}+\frac{3}{4}c^{16}_{0,3}+\frac{1}{2}c^{16}_{0,4}+\frac{5}{16}c^{16}_{0,5}\right)=0
\end{aligned}
$$

The constrains corresponding to φ_{17}:

$$
\begin{aligned}
\varphi_{17}|_{(0,0)} &= c^{17}_{0,0}=0\\
\varphi_{17}|_{(1,0)} &= c^{17}_{0,0}+c^{17}_{1,0}+c^{17}_{2,0}+c^{17}_{3,0}+c^{17}_{4,0}+c^{17}_{5,0}=0\\
\varphi_{17}|_{(0,1)} &= c^{17}_{0,0}+c^{17}_{0,1}+c^{17}_{0,2}+c^{17}_{0,3}+c^{17}_{0,4}+c^{17}_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial\varphi_{17}}{\partial\xi}\Big|_{(0,0)} &= c^{17}_{1,0}=0\\
\frac{\partial\varphi_{17}}{\partial\xi}\Big|_{(1,0)} &= c^{17}_{1,0}+2c^{17}_{2,0}+3c^{17}_{3,0}+4c^{17}_{4,0}+5c^{17}_{5,0}=0\\
\frac{\partial\varphi_{17}}{\partial\xi}\Big|_{(0,1)} &= c^{17}_{1,0}+c^{17}_{1,1}\eta+c^{17}_{1,2}+c^{17}_{1,2}+c^{17}_{1,4}=0\\
\frac{\partial\varphi_{17}}{\partial\eta}\Big|_{(0,0)} &= c^{17}_{0,1}=0\\
\frac{\partial\varphi_{17}}{\partial\eta}\Big|_{(1,0)} &= c^{17}_{0,1}+c^{17}_{1,1}+c^{17}_{2,1}+c^{17}_{3,1}+c^{17}_{4,1}=0\\
\frac{\partial\varphi_{17}}{\partial\eta}\Big|_{(0,1)} &= c^{17}_{0,1}+2c^{17}_{0,2}+3c^{17}_{0,3}+4c^{17}_{0,4}+5c^{17}_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2\varphi_{17}}{\partial\xi^2}\right|_{(0,0)} &= 2c^{17}_{2,0} = 0\\
\left.\frac{\partial^2\varphi_{17}}{\partial\xi^2}\right|_{(1,0)} &= 2c^{17}_{2,0} + 6c^{17}_{3,0} + 12c^{17}_{4,0} + 20c^{17}_{5,0} = 0\\
\left.\frac{\partial^2\varphi_{17}}{\partial\xi^2}\right|_{(0,1)} &= 2c^{17}_{2,0} + 2c^{17}_{2,1} + 2c^{17}_{2,2} + 2c^{17}_{2,3} = 0\\
\left.\frac{\partial^2\varphi_{17}}{\partial\xi\partial\eta}\right|_{(0,0)} &= c^{17}_{1,1} = 0\\
\left.\frac{\partial^2\varphi_{17}}{\partial\xi\partial\eta}\right|_{(1,0)} &= c^{17}_{1,1} + 2c^{17}_{2,1} + 3c^{17}_{3,1} + 4c^{17}_{4,1} = 0\\
\left.\frac{\partial^2\varphi_{17}}{\partial\xi\partial\eta}\right|_{(0,1)} &= c^{17}_{1,1} + 2c^{17}_{1,2} + 3c^{17}_{1,2} + 4c^{17}_{1,4} = 0\\
\left.\frac{\partial^2\varphi_{0}}{\partial\eta^2}\right|_{(0,0)} &= 2c^{17}_{0,2} = 0\\
\left.\frac{\partial^2\varphi_{17}}{\partial\eta^2}\right|_{(1,0)} &= 2c^{17}_{0,2} + 2c^{17}_{1,2} + 2c^{17}_{2,2} + 2c^{17}_{3,2} = 0\\
\left.\frac{\partial^2\varphi_{17}}{\partial\eta^2}\right|_{(0,1)} &= 2c^{17}_{0,2} + 6c^{17}_{0,3} + 12c^{17}_{0,4} + 20c^{17}_{0,5} = 1
\end{aligned}
$$

$$
\begin{aligned}
\left.\left(\frac{\partial\varphi_{17}}{\partial\xi}n_\xi + \frac{\partial\varphi_{17}}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_{17}}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^{17}_{0,1} + \frac{1}{2}c^{17}_{1,1} + \frac{1}{4}c^{17}_{2,1} + \frac{1}{8}c^{17}_{3,1} + \frac{1}{16}c^{17}_{4,1} = 0\\
\left.\left(\frac{\partial\varphi_{17}}{\partial\xi}n_\xi + \frac{\partial\varphi_{17}}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{17}_{1,0} + c^{17}_{2,0} + c^{17}_{1,1} + \frac{3}{4}c^{17}_{3,0} + \frac{3}{4}c^{17}_{2,1} + \frac{3}{4}c^{17}_{1,2} + \frac{1}{2}c^{17}_{4,0} +\\
&\quad \frac{1}{2}c^{17}_{3,1} + \frac{1}{2}c^{17}_{2,2} + \frac{1}{2}c^{17}_{1,2} + \frac{5}{16}c^{17}_{5,0} + \frac{9}{16}c^{17}_{4,1} + \frac{7}{16}c^{17}_{3,2} +\\
&\quad \frac{7}{16}c^{17}_{2,3} + \frac{5}{16}c^{17}_{1,4} + c^{17}_{0,1} + c^{17}_{0,2} + \frac{3}{4}c^{17}_{0,3} + \frac{1}{2}c^{17}_{0,4} + \frac{5}{16}c^{17}_{0,5}) = 0\\
\left.\left(\frac{\partial\varphi_{17}}{\partial\xi}n_\xi + \frac{\partial\varphi_{17}}{\partial\eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_{17}}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^{17}_{0,1} + c^{17}_{0,2} + \frac{3}{4}c^{17}_{0,3} + \frac{1}{2}c^{17}_{0,4} + \frac{5}{16}c^{17}_{0,5}\right) = 0
\end{aligned}
$$

The constrains corresponding to φ_{18}:

$$
\begin{aligned}
\varphi_{18}|_{(0,0)} &= c^{18}_{0,0} = 0\\
\varphi_{18}|_{(1,0)} &= c^{18}_{0,0} + c^{18}_{1,0} + c^{18}_{2,0} + c^{18}_{3,0} + c^{18}_{4,0} + c^{18}_{5,0} = 0\\
\varphi_{18}|_{(0,1)} &= c^{18}_{0,0} + c^{18}_{0,1} + c^{18}_{0,2} + c^{18}_{0,3} + c^{18}_{0,4} + c^{18}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_{18}}{\partial \xi}\right|_{(0,0)} &= c_{1,0}^{18} = 0 \\
\left.\frac{\partial \varphi_{18}}{\partial \xi}\right|_{(1,0)} &= c_{1,0}^{18} + 2c_{2,0}^{18} + 3c_{3,0}^{18} + 4c_{4,0}^{18} + 5c_{5,0}^{18} = 0 \\
\left.\frac{\partial \varphi_{18}}{\partial \xi}\right|_{(0,1)} &= c_{1,0}^{18} + c_{1,1}^{18}\eta + c_{1,2}^{18} + c_{1,2}^{18} + c_{1,4}^{18} = 0 \\
\left.\frac{\partial \varphi_{18}}{\partial \eta}\right|_{(0,0)} &= c_{0,1}^{18} = 0 \\
\left.\frac{\partial \varphi_{18}}{\partial \eta}\right|_{(1,0)} &= c_{0,1}^{18} + c_{1,1}^{18} + c_{2,1}^{18} + c_{3,1}^{18} + c_{4,1}^{8} = 0 \\
\left.\frac{\partial \varphi_{18}}{\partial \eta}\right|_{(0,1)} &= c_{0,1}^{18} + 2c_{0,2}^{18} + 3c_{0,3}^{18} + 4c_{0,4}^{18} + 5c_{0,5}^{18} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_{18}}{\partial \xi^2}\right|_{(0,0)} &= 2c_{2,0}^{18} = 0 \\
\left.\frac{\partial^2 \varphi_{18}}{\partial \xi^2}\right|_{(1,0)} &= 2c_{2,0}^{18} + 6c_{3,0}^{18} + 12c_{4,0}^{18} + 20c_{5,0}^{18} = 0 \\
\left.\frac{\partial^2 \varphi_{18}}{\partial \xi^2}\right|_{(0,1)} &= 2c_{2,0}^{18} + 2c_{2,1}^{18} + 2c_{2,2}^{18} + 2c_{2,3}^{18} = 0 \\
\left.\frac{\partial^2 \varphi_{18}}{\partial \xi \partial \eta}\right|_{(0,0)} &= c_{1,1}^{18} = 0 \\
\left.\frac{\partial^2 \varphi_{18}}{\partial \xi \partial \eta}\right|_{(1,0)} &= c_{1,1}^{18} + 2c_{2,1}^{18} + 3c_{3,1}^{18} + 4c_{4,1}^{18} = 0 \\
\left.\frac{\partial^2 \varphi_{18}}{\partial \xi \partial \eta}\right|_{(0,1)} &= c_{1,1}^{18} + 2c_{1,2}^{18} + 3c_{1,2}^{18} + 4c_{1,4}^{18} = 0 \\
\left.\frac{\partial^2 \varphi_{18}}{\partial \eta^2}\right|_{(0,0)} &= 2c_{0,2}^{18} = 0 \\
\left.\frac{\partial^2 \varphi_{18}}{\partial \eta^2}\right|_{(1,0)} &= 2c_{0,2}^{18} + 2c_{1,2}^{18} + 2c_{2,2}^{18} + 2c_{3,2}^{18} = 0 \\
\left.\frac{\partial^2 \varphi_{18}}{\partial \eta^2}\right|_{(0,1)} &= 2c_{0,2}^{18} + 6c_{0,3}^{18} + 12c_{0,4}^{18} + 20c_{0,5}^{18} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left(\frac{\partial\varphi_{18}}{\partial\xi}n_\xi+\frac{\partial\varphi_{18}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} &= \frac{\partial\varphi_{18}}{\partial\eta}\Big|_{\left(\frac{1}{2},0\right)} = c^{18}_{0,1}+\frac{1}{2}c^{18}_{1,1}+\frac{1}{4}c^{18}_{2,1}+\frac{1}{8}c^{18}_{3,1}+\frac{1}{16}c^{18}_{4,1}=1\\
\left(\frac{\partial\varphi_{18}}{\partial\xi}n_\xi+\frac{\partial\varphi_{18}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{18}_{1,0}+c^{18}_{2,0}+c^{18}_{1,1}+\frac{3}{4}c^{18}_{3,0}+\frac{3}{4}c^{18}_{2,1}+\frac{3}{4}c^{18}_{1,2}+\frac{1}{2}c^{18}_{4,0}+\\
&\quad \frac{1}{2}c^{18}_{3,1}+\frac{1}{2}c^{18}_{2,2}+\frac{1}{2}c^{18}_{1,2}+\frac{5}{16}c^{18}_{5,0}+\frac{9}{16}c^{18}_{4,1}+\frac{7}{16}c^{18}_{3,2}+\\
&\quad \frac{7}{16}c^{18}_{2,3}+\frac{5}{16}c^{18}_{1,4}+c^{18}_{0,1}+c^{18}_{0,2}+\frac{3}{4}c^{18}_{0,3}+\frac{1}{2}c^{18}_{0,4}+\frac{5}{16}c^{18}_{0,5})=0\\
\left(\frac{\partial\varphi_{18}}{\partial\xi}n_\xi+\frac{\partial\varphi_{18}}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} &= -\frac{\partial\varphi_{18}}{\partial\xi}\Big|_{\left(0,\frac{1}{2}\right)} = -\left(c^{18}_{0,1}+c^{18}_{0,2}+\frac{3}{4}c^{18}_{0,3}+\frac{1}{2}c^{18}_{0,4}+\frac{5}{16}c^{18}_{0,5}\right)=0
\end{aligned}
$$

The constrains corresponding to φ_{19}:

$$
\begin{aligned}
\varphi_{19}|_{(0,0)} &= c^{19}_{0,0}=0\\
\varphi_{19}|_{(1,0)} &= c^{19}_{0,0}+c^{19}_{1,0}+c^{19}_{2,0}+c^{19}_{3,0}+c^{19}_{4,0}+c^{19}_{5,0}=0\\
\varphi_{19}|_{(0,1)} &= c^{19}_{0,0}+c^{19}_{0,1}+c^{19}_{0,2}+c^{19}_{0,3}+c^{19}_{0,4}+c^{19}_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial\varphi_{19}}{\partial\xi}\Big|_{(0,0)} &= c^{19}_{1,0}=0\\
\frac{\partial\varphi_{19}}{\partial\xi}\Big|_{(1,0)} &= c^{19}_{1,0}+2c^{19}_{2,0}+3c^{19}_{3,0}+4c^{19}_{4,0}+5c^{19}_{5,0}=0\\
\frac{\partial\varphi_{19}}{\partial\xi}\Big|_{(0,1)} &= c^{19}_{1,0}+c^{19}_{1,1}\eta+c^{19}_{1,2}+c^{19}_{1,2}+c^{19}_{1,4}=0\\
\frac{\partial\varphi_{19}}{\partial\eta}\Big|_{(0,0)} &= c^{19}_{0,1}=0\\
\frac{\partial\varphi_{19}}{\partial\eta}\Big|_{(1,0)} &= c^{19}_{0,1}+c^{19}_{1,1}+c^{19}_{2,1}+c^{19}_{3,1}+c^{19}_{4,1}=0\\
\frac{\partial\varphi_{19}}{\partial\eta}\Big|_{(0,1)} &= c^{19}_{0,1}+2c^{19}_{0,2}+3c^{19}_{0,3}+4c^{19}_{0,4}+5c^{19}_{0,5}=0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2\varphi_{19}}{\partial\xi^2}\right|_{(0,0)} &= 2c^{19}_{2,0} = 0\\
\left.\frac{\partial^2\varphi_{19}}{\partial\xi^2}\right|_{(1,0)} &= 2c^{19}_{2,0} + 6c^{19}_{3,0} + 12c^{19}_{4,0} + 20c^{19}_{5,0} = 0\\
\left.\frac{\partial^2\varphi_{19}}{\partial\xi^2}\right|_{(0,1)} &= 2c^{19}_{2,0} + 2c^{19}_{2,1} + 2c^{19}_{2,2} + 2c^{19}_{2,3} = 0\\
\left.\frac{\partial^2\varphi_{19}}{\partial\xi\partial\eta}\right|_{(0,0)} &= c^{19}_{1,1} = 0\\
\left.\frac{\partial^2\varphi_{19}}{\partial\xi\partial\eta}\right|_{(1,0)} &= c^{19}_{1,1} + 2c^{19}_{2,1} + 3c^{19}_{3,1} + 4c^{19}_{4,1} = 0\\
\left.\frac{\partial^2\varphi_{19}}{\partial\xi\partial\eta}\right|_{(0,1)} &= c^{19}_{1,1} + 2c^{19}_{1,2} + 3c^{19}_{1,2} + 4c^{19}_{1,4} = 0\\
\left.\frac{\partial^2\varphi_{19}}{\partial\eta^2}\right|_{(0,0)} &= 2c^{19}_{0,2} = 0\\
\left.\frac{\partial^2\varphi_{19}}{\partial\eta^2}\right|_{(1,0)} &= 2c^{19}_{0,2} + 2c^{19}_{1,2} + 2c^{19}_{2,2} + 2c^{19}_{3,2} = 0\\
\left.\frac{\partial^2\varphi_{19}}{\partial\eta^2}\right|_{(0,1)} &= 2c^{19}_{0,2} + 6c^{19}_{0,3} + 12c^{19}_{0,4} + 20c^{19}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\left(\frac{\partial\varphi_{19}}{\partial\xi}n_\xi + \frac{\partial\varphi_{19}}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},0\right)} &= \left.\frac{\partial\varphi_{19}}{\partial\eta}\right|_{\left(\frac{1}{2},0\right)} = c^{19}_{0,1} + \frac{1}{2}c^{19}_{1,1} + \frac{1}{4}c^{19}_{2,1} + \frac{1}{8}c^{19}_{3,1} + \frac{1}{16}c^{19}_{4,1} = 0\\
\left.\left(\frac{\partial\varphi_{19}}{\partial\xi}n_\xi + \frac{\partial\varphi_{19}}{\partial\eta}n_\eta\right)\right|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{19}_{1,0} + c^{19}_{2,0} + c^{19}_{1,1} + \frac{3}{4}c^{19}_{3,0} + \frac{3}{4}c^{19}_{2,1} + \frac{3}{4}c^{19}_{1,2} + \frac{1}{2}c^{19}_{4,0} +\\
&\quad \frac{1}{2}c^{19}_{3,1} + \frac{1}{2}c^{19}_{2,2} + \frac{1}{2}c^{19}_{1,2} + \frac{5}{16}c^{19}_{5,0} + \frac{9}{16}c^{19}_{4,1} + \frac{7}{16}c^{19}_{3,2} +\\
&\quad \frac{7}{16}c^{19}_{2,3} + \frac{5}{16}c^{19}_{1,4} + c^{19}_{0,1} + c^{19}_{0,2} + \frac{3}{4}c^{19}_{0,3} + \frac{1}{2}c^{19}_{0,4} + \frac{5}{16}c^{19}_{0,5}) = 1\\
\left.\left(\frac{\partial\varphi_{19}}{\partial\xi}n_\xi + \frac{\partial\varphi_{19}}{\partial\eta}n_\eta\right)\right|_{\left(0,\frac{1}{2}\right)} &= -\left.\frac{\partial\varphi_{19}}{\partial\xi}\right|_{\left(0,\frac{1}{2}\right)} = -\left(c^{19}_{0,1} + c^{19}_{0,2} + \frac{3}{4}c^{19}_{0,3} + \frac{1}{2}c^{19}_{0,4} + \frac{5}{16}c^{19}_{0,5}\right) = 0
\end{aligned}
$$

The constrains corresponding to φ_{20}:

$$
\begin{aligned}
\varphi_{20}|_{(0,0)} &= c^{20}_{0,0} = 0\\
\varphi_{20}|_{(1,0)} &= c^{20}_{0,0} + c^{20}_{1,0} + c^{20}_{2,0} + c^{20}_{3,0} + c^{20}_{4,0} + c^{20}_{5,0} = 0\\
\varphi_{20}|_{(0,1)} &= c^{20}_{0,0} + c^{20}_{0,1} + c^{20}_{0,2} + c^{20}_{0,3} + c^{20}_{0,4} + c^{20}_{0,5} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial \varphi_{20}}{\partial \xi}\right|_{(0,0)} &= c_{1,0}^{20} = 0 \\
\left.\frac{\partial \varphi_{20}}{\partial \xi}\right|_{(1,0)} &= c_{1,0}^{20} + 2c_{2,0}^{20} + 3c_{3,0}^{20} + 4c_{4,0}^{20} + 5c_{5,0}^{20} = 0 \\
\left.\frac{\partial \varphi_{20}}{\partial \xi}\right|_{(0,1)} &= c_{1,0}^{20} + c_{1,1}^{20}\eta + c_{1,2}^{20} + c_{1,2}^{20} + c_{1,4}^{20} = 0 \\
\left.\frac{\partial \varphi_{20}}{\partial \eta}\right|_{(0,0)} &= c_{0,1}^{20} = 0 \\
\left.\frac{\partial \varphi_{20}}{\partial \eta}\right|_{(1,0)} &= c_{0,1}^{20} + c_{1,1}^{20} + c_{2,1}^{20} + c_{3,1}^{20} + c_{4,1}^{20} = 0 \\
\left.\frac{\partial \varphi_{20}}{\partial \eta}\right|_{(0,1)} &= c_{0,1}^{20} + 2c_{0,2}^{20} + 3c_{0,3}^{20} + 4c_{0,4}^{20} + 5c_{0,5}^{20} = 0
\end{aligned}
$$

$$
\begin{aligned}
\left.\frac{\partial^2 \varphi_{20}}{\partial \xi^2}\right|_{(0,0)} &= 2c_{2,0}^{20} = 0 \\
\left.\frac{\partial^2 \varphi_{20}}{\partial \xi^2}\right|_{(1,0)} &= 2c_{2,0}^{20} + 6c_{3,0}^{20} + 12c_{4,0}^{20} + 20c_{5,0}^{20} = 0 \\
\left.\frac{\partial^2 \varphi_{20}}{\partial \xi^2}\right|_{(0,1)} &= 2c_{2,0}^{20} + 2c_{2,1}^{20} + 2c_{2,2}^{20} + 2c_{2,3}^{20} = 0 \\
\left.\frac{\partial^2 \varphi_{20}}{\partial \xi \partial \eta}\right|_{(0,0)} &= c_{1,1}^{20} = 0 \\
\left.\frac{\partial^2 \varphi_{20}}{\partial \xi \partial \eta}\right|_{(1,0)} &= c_{1,1}^{20} + 2c_{2,1}^{20} + 3c_{3,1}^{20} + 4c_{4,1}^{20} = 0 \\
\left.\frac{\partial^2 \varphi_{20}}{\partial \xi \partial \eta}\right|_{(0,1)} &= c_{1,1}^{20} + 2c_{1,2}^{20} + 3c_{1,2}^{20} + 4c_{1,4}^{20} = 0 \\
\left.\frac{\partial^2 \varphi_{20}}{\partial \eta^2}\right|_{(0,0)} &= 2c_{0,2}^{20} = 0 \\
\left.\frac{\partial^2 \varphi_{20}}{\partial \eta^2}\right|_{(1,0)} &= 2c_{0,2}^{20} + 2c_{1,2}^{20} + 2c_{2,2}^{20} + 2c_{3,2}^{20} = 0 \\
\left.\frac{\partial^2 \varphi_{20}}{\partial \eta^2}\right|_{(0,1)} &= 2c_{0,2}^{20} + 6c_{0,3}^{20} + 12c_{0,4}^{20} + 20c_{0,5}^{20} = 0
\end{aligned}
$$

$$\left(\frac{\partial\varphi_{20}}{\partial\xi}n_\xi+\frac{\partial\varphi_{20}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},0\right)} = \frac{\partial\varphi_{20}}{\partial\eta}\Big|_{\left(\frac{1}{2},0\right)} = c^{20}_{0,1}+\frac{1}{2}c^{20}_{1,1}+\frac{1}{4}c^{20}_{2,1}+\frac{1}{8}c^{20}_{3,1}+\frac{1}{16}c^{20}_{4,1}=0$$

$$\begin{aligned}\left(\frac{\partial\varphi_{20}}{\partial\xi}n_\xi+\frac{\partial\varphi_{20}}{\partial\eta}n_\eta\right)\Big|_{\left(\frac{1}{2},\frac{1}{2}\right)} &= -\frac{\sqrt{2}}{2}(c^{20}_{1,0}+c^{20}_{2,0}+c^{20}_{1,1}+\frac{3}{4}c^{20}_{3,0}+\frac{3}{4}c^{20}_{2,1}+\frac{3}{4}c^{20}_{1,2}+\frac{1}{2}c^{20}_{4,0}+\\ &\quad \frac{1}{2}c^{20}_{3,1}+\frac{1}{2}c^{20}_{2,2}+\frac{1}{2}c^{20}_{1,2}+\frac{5}{16}c^{20}_{5,0}+\frac{9}{16}c^{20}_{4,1}+\frac{7}{16}c^{20}_{3,2}+\\ &\quad \frac{7}{16}c^{20}_{2,3}+\frac{5}{16}c^{20}_{1,4}+c^{20}_{0,1}+c^{20}_{0,2}+\frac{3}{4}c^{20}_{0,3}+\frac{1}{2}c^{20}_{0,4}+\frac{5}{16}c^{20}_{0,5})=0\end{aligned}$$

$$\left(\frac{\partial\varphi_{20}}{\partial\xi}n_\xi+\frac{\partial\varphi_{20}}{\partial\eta}n_\eta\right)\Big|_{\left(0,\frac{1}{2}\right)} = -\frac{\partial\varphi_{20}}{\partial\xi}\Big|_{\left(0,\frac{1}{2}\right)} = -\left(c^{20}_{0,1}+c^{20}_{0,2}+\frac{3}{4}c^{20}_{0,3}+\frac{1}{2}c^{20}_{0,4}+\frac{5}{16}c^{20}_{0,5}\right)=1$$

Appendix B: Transforming the Non-conserved Adjoint Euler Equation to Conserved Equation

Consider the linearized steady-state Euler equations:

$$LU = \frac{\partial}{\partial x}\left(A_x U\right) + \frac{\partial}{\partial y}\left(A_y U\right) = f \tag{1}$$

where

$$A_x = \frac{\partial F_x}{\partial U} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ \frac{\gamma-1}{2}\left(u^2+v^2\right) - u^2 & \left(3-\gamma\right)u & \left(1-\gamma\right)v & \gamma - 1 \\ -uv & v & u & 0 \\ u\left(\frac{\gamma-1}{2}\left(u^2+v^2\right) - h\right) & h - \left(\gamma-1\right)u^2 & \left(1-\gamma\right)uv & \gamma u \end{pmatrix}$$

$$A_y = \frac{\partial F_y}{\partial U} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ -uv & v & u & 0 \\ \frac{\gamma-1}{2}\left(u^2+v^2\right) - v^2 & \left(1-\gamma\right)u & \left(3-\gamma\right)v & \gamma - 1 \\ v\left(\frac{\gamma-1}{2}\left(u^2+v^2\right) - h\right) & \left(1-\gamma\right)uv & h - \left(\gamma-1\right)v^2 & \gamma v \end{pmatrix}$$

To obtain the continuous adjoint equation, Eq. 1 is multiplied by Z, integrated over the domain D and integrated by parts:

$$\begin{aligned} \left(\frac{\partial}{\partial x}\left(A_x U\right) + \frac{\partial}{\partial y}\left(A_y U\right), Z\right)_D &= \left(U, -A_x^T\frac{\partial Z}{\partial x} - A_y^T\frac{\partial Z}{\partial y}\right)_D \\ &+ \left(\left(n_x A_x + n_y A_y\right)U, Z\right)_{\partial D}. \end{aligned} \tag{2}$$

We end up with $L^*Z = -A_x^T \frac{\partial Z}{\partial x} - A_y^T \frac{\partial Z}{\partial y}$ for the continuous adjoint problem which is not in conserved form. The left-hand side can be written as:

$$
-\frac{\partial}{\partial x}\begin{pmatrix} \left(\frac{(\gamma-1)}{2}\left(u^2+v^2\right)-u^2\right)Z_2 - uvZ_3 + u\left(\frac{\gamma-1}{2}\left(u^2+v^2\right)-h\right)Z_4 \\ Z_1 + (3-\gamma)\,uZ_2 + vZ_3 + \left(h-(\gamma-1)\,u^2\right)Z_4 \\ (1-\gamma)\,vZ_2 + uZ_3 + (1-\gamma)\,uvZ_4 \\ (\gamma-1)\,Z_2 + \gamma u Z_4 \end{pmatrix}
$$

$$
-\frac{\partial}{\partial y}\begin{pmatrix} -uvZ_2 + \left(\frac{\gamma-1}{2}\left(u^2+v^2\right)-v^2\right)Z_3 + v\left(\frac{\gamma-1}{2}\left(u^2+v^2\right)-h\right)Z_4 \\ vZ_2 + (1-\gamma)\,uZ_3 + (1-\gamma)\,uvZ_4 \\ Z_1 + uZ_2 + (3-\gamma)\,vZ_3 + \left(h-(\gamma-1)\,v^2\right)Z_4 \\ (\gamma-1)\,Z_3 + \gamma v Z_4 \end{pmatrix}
$$

$$
+\begin{pmatrix} Z_2\frac{\partial}{\partial x}\left(\frac{(\gamma-1)}{2}\left(u^2+v^2\right)-u^2\right) - Z_3\frac{\partial}{\partial x}(uv) + Z_4\frac{\partial}{\partial x}\left(u\left(\frac{\gamma-1}{2}\left(u^2+v^2\right)-h\right)\right) \\ Z_2\frac{\partial}{\partial x}(3-\gamma)\,u + Z_3\frac{\partial}{\partial x}(v) + Z_4\frac{\partial}{\partial x}\left(h-(\gamma-1)\,u^2\right) \\ Z_2\frac{\partial}{\partial x}\left((1-\gamma)\,v\right) + Z_3\frac{\partial}{\partial x}(u) + Z_4\frac{\partial}{\partial x}\left((1-\gamma)\,uv\right) \\ Z_2\frac{\partial}{\partial x}(\gamma-1) + Z_4\frac{\partial}{\partial x}(\gamma u) \end{pmatrix}
$$

$$
+\begin{pmatrix} -Z_2\frac{\partial}{\partial y}(uv) + Z_3\frac{\partial}{\partial y}\left(\frac{\gamma-1}{2}\left(u^2+v^2\right)-v^2\right) + Z_4\frac{\partial}{\partial y}\left(v\left(\frac{\gamma-1}{2}\left(u^2+v^2\right)-h\right)\right) \\ Z_2\frac{\partial}{\partial y}(v) + Z_3\frac{\partial}{\partial y}\left((1-\gamma)\,u\right) + Z_4\frac{\partial}{\partial y}\left((1-\gamma)\,uv\right) \\ Z_2\frac{\partial}{\partial y}(u) + Z_3\frac{\partial}{\partial y}\left((3-\gamma)\,v\right) + Z_4\frac{\partial}{\partial y}\left(h-(\gamma-1)\,v^2\right) \\ Z_3\frac{\partial}{\partial y}(\gamma-1) + Z_4\frac{\partial}{\partial y}(\gamma v) \end{pmatrix}
$$

where the first two terms are now in conserved form and the last two terms are considered as the source term.