Mesh Adaptation for Wakes via Surface Insertion

by

Shahzaib Malik

BEng., Queen Mary University of London (UK), 2013

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

MASTER OF APPLIED SCIENCE

in

The Faculty of Graduate and Postdoctoral Studies

(Mechanical Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

May 2019

The following individuals certify that they have read, and recommend to the Faculty of Graduate and Postdoctoral Studies for acceptance, a thesis/dissertation entitled:

Mesh Adaptation for Wakes via Surface Insertion

submitted by Shahzaib Malik in partial fulfillment of the requirements for

the degree of Master of Applied Science

in Mechanical Engineering

Examining Committee:

Dr. Carl Ollivier-Gooch
Supervisor

Dr. Dana Grecov
Supervisory Committee Member

Dr. Steven Rogak
Supervisory Committee Member

Abstract

In this thesis, we present a new procedure for mesh adaptation for wakes. The approach starts by tracking the wake centerline with an initial isotropic unstructured mesh. A vertex-centered finite volume method is used, and the velocity field is obtained from solution reconstruction. The velocity data is integrated numerically using an adaptive fourth-order Runge-Kutta method. We insert the wake centerline into the existing unstructured mesh as an internal boundary and use a metric-based anisotropic mesh adaptation to generate anisotropic cells in regions with large second derivatives of flow variables. In the second step, the problem is solved on adapted mesh and a new wake centerline is tracked. We then move the previous wake centerline (which is now a part of adapted mesh) to match the centerline obtained from the adapted mesh. To move the wake centerline, a solid mechanics analogy is used and the linear elasticity equation is solved on the adapted mesh. As a result, the displacement is propagated throughout the mesh and the already adapted regions along the wake centerline are preserved. The process is then followed for subsequent cycles of anisotropic mesh adaptation to obtain a more accurate approximation of the wake centerline.

As an alternate strategy for obtaining an anisotropic mesh in the wake, we take the first geometry, together with the captured wake centerline from an unstructured triangular mesh, as an initial geometry to produce a quad dominant mesh, using an advancing layer method.

The correctness of the streamline tracking algorithm is verified using an analytical velocity field. The mesh morphing approach is tested using the method of manufactured solutions, demonstrating that the linear finite element solution is second-order accurate. The results of laminar flow test cases for the attached and separated flow are presented and compared with some well-established numerical results in the literature. Our results show that the advancing layer mesh is more efficient in resolving the wake. In the end, one case for turbulent subsonic flow is considered. For turbulent flow, a cell-centered finite volume method is used and we only track the wake centerline at different angles of attack.

Lay Summary

With the advancements in computing resources, it has become important to develop CFD methods in such a way that they are efficient in terms of computational cost while maintaining and improving the solution accuracy at the same time. The errors in numerical simulations can influence the flow physics and solution accuracy on different scales and are highly dependent on mesh spacing. Mesh refinement can help to reduce these errors but uniform mesh refinement is not computationally efficient. Many researchers focus on selectively refining the mesh in areas where the impact of errors on the solution is greater and such a process is known as solution adaptive refinement. This research introduces a new approach to anisotropic mesh adaptation for wakes. The results from different meshes are compared to identify the more efficient mesh in tracking the wake.

Preface

The research work presented in this thesis is the outcome of a close working relationship between Dr. Carl Ollivier Gooch and Shahzaib Malik. The algorithm development, concept formulation, data analysis, and manuscript preparation were done by the author, Shahzaib Malik, under valuable guidance from Dr. Carl Ollivier Gooch in the department of Mechanical Engineering at the University of British Columbia.

Some of the results in Chapter 4 of this thesis were presented by the author at the AIAA SciTech 2019 Forum in January 2019. Parts of Chapter 1, Chapter 3 and Chapter 4 were published in the paper [31]: "Shahzaib Malik and Carl F Ollivier Gooch. Mesh adaptation for wakes via surface insertion. In *AIAA SciTech 2019 Forum*, AIAA Paper 2019-1996, 2019". `https://arc.aiaa.org/doi/abs/10.2514/6.2019-1996`

Table of Contents

List of Tables

List of Figures

Nomenclature

Roman Symbols

AR	aspect ratio
c	chord length
C_D	drag coefficient
C_L	lift coefficient
d	distance function, linear elasticity coefficients
dy_{min}	off-wall spacing
E	Young's modulus
F	flux dyad
H	Hessian matrix
L	deformation tensor
l_M	metric length
l_M^{max}	maximum metric length
l_M^{min}	minimum metric length
M	metric
Ma	Mach number

Re	Reynolds number
R	rotation matrix
S	source term
U	solution vector
u	x-velocity
v	y-velocity

Greek Symbols

α	angle of attack
Λ	size matrix
ν	Poisson's ratio

Acknowledgements

First of all, I would like to thank my supervisor Dr. Carl Ollivier Gooch for his valuable time, supervision and constant support throughout my research and graduate studies. To be honest, Carl's enthusiasm for research and involvement with research students fueled me during my time at UBC. I am glad that I worked with him and learned so many things under his supervision.

I would like to express my gratitude to ANSYS Canada, the Natural Sciences and Engineering Research Council of Canada and the University of British Columbia for providing financial support.

I am thankful to the members of the supervisory committee: Dr. Steven Rogak and Dr. Dana Grecov, for generously offering their time to review my thesis.

Also, I am very thankful to all my colleagues at Advanced Numerical Simulation Lab: Gary, Reza, Jasmeet, Ehsan and Akhil for their encouragement and valuable discussions regarding the challenges faced during the research work.

Finally, thanks to my friends: Faisal, Zhihao, Somesh, Harsh, Claire, Juuso, Alondra, Miguel, Adriana, Ratul and Sarah for being with me.

Dedication

I would like to thank my family especially my mom, dad and sister as their support, trust and encouragement has always made be strong enough to overcome all the challenges.

To my family.

Chapter 1

Introduction

Computational fluid dynamics (CFD) has become an increasingly valuable tool in the last few decades with many industrial and non-industrial applications. These applications include aerodynamics of motor vehicles and aircraft, oil recovery, turbo-machinery, hydrodynamics of ships and astrophysics [9, 11, 57]. Therefore, it is immensely important to further develop CFD methods in such a way that they are efficient in terms of computational cost while maintaining and improving the solution accuracy at the same time. One of the main objectives of CFD analysis is to study the physical processes that occur around different objects. These processes can be the result of phenomena such as shock waves, boundary layers, flow separation and turbulence. The processes are governed by different equations depending on the type of problem under investigation. It is possible to obtain an analytical solution to many conservation equations but only under certain simple conditions. Analytical solutions are usually possible when the equation is linear and the geometry and boundary conditions are not very complicated. In reality, most of the problems in fluid dynamics are governed by the Navier-Stokes equations as the effect of viscosity cannot be ignored very close to the boundary when drag or heat transfer is important. The Navier-Stokes equations are non-linear and very complicated to solve, thus making analytical solutions impossible especially for complex geometries and this is where numerical methods become very useful. Different numerical methods exist that can be used to obtain numerical solutions to the ordinary and partial differential equations related to the problem [10, 28].

When numerical methods are applied to differential equations, the original equations are discretized and the resulting algebraic equations are then solved through numerical methods. As a result of this discretization, the numerical solution obtained is an approximate solution rather than the exact solution. The difference between the exact and numerical solution is the numerical error. These errors can effects flow physics and solution on different scales and are highly dependent on the mesh spacing. One way to reduce these errors is to reduce the mesh spacing uniformly and ideally refine the mesh until the solution is grid converged. However, uniform mesh refinement is not computationally efficient and usually, the focus is to identify those areas where the impact of errors on the solution is greater and then reduce the error in those areas by refining the mesh locally. Refining the mesh in such a way is known as solution adaptive refinement [17, 43].

This thesis deals with anisotropic mesh adaptation which is one step in the relentless effort to make more efficient use of computing resources.

1.1 Motivation and Objectives

CFD analysis to obtain a numerical solution involves three essential steps: (1) modeling, (2) mesh generation and (3) solution. Modeling refers to specifying the problem geometry, governing equations and the boundary conditions. In the second step, to solve the governing equations numerically on the specified geometry, the continuous domain is divided into a set of discrete regions known as the mesh. The third step involves discretization of the equations on the mesh and solution of the resulting system of algebraic equations to obtain the numerical solution [54]. For real aerodynamics applications and industry level problems, the mesh generation process is considered to be the most time consuming out of all processes in terms of human time and can be up to 45 times longer compared to the solver time [2, 33]. Therefore, it is very important to explore ways to reduce the mesh generation time for a given level of solution accuracy.

A mesh is a discrete representation of the geometry of a CFD problem and has significant effects on convergence, computational cost and accuracy of the numerical solution [13, 54]. Therefore, it is immensely important to obtain a good quality mesh and take into consideration the geometric properties of the mesh such as aspect ratio, smoothness, maximum and minimum angle, local mesh size and orientation. Any poor quality cells can adversely affect the numerical approximation and thus lead to unreliable results and instability in the solution. There is generally a trade-off between the accuracy and computational cost in computational fluid dynamics. In many aerodynamics applications, accuracy can be improved for a given computational cost by using high aspect ratio cells to resolve the flow properties. A typical example is a boundary layer, where there are high velocity gradients perpendicular to the wall and small gradients parallel to the wall. Obtaining a reasonable accuracy in the boundary layer using fine isotropic cells leads to additional memory usage and unnecessary computational effort. Therefore, it is more efficient to use anisotropic cells with anisotropy aligned along the desired direction as the anisotropic cells help to improve the inverse relationship between solution accuracy and computational cost. Many researchers such as Bottaso [6], Huang [18], Apel [4] and Loseille [29] have provided a detailed view on the advantages of using anisotropic elements and metric-based anisotropic mesh adaptation schemes. Moreover, Rippa [45] showed that for linear reconstruction, the cells should have longer edge length in the direction where the second derivatives of solution variables are small compared to the direction with large second derivatives [49].

Why do we care about resolving the wake?

A wake is the region of decelerated flow behind a body immersed in a fluid [59]. It can be thought of as the boundary layer separated from the rear of the body. Quite often in aerodynamics, we study the flow over individual components such as wing, fuselage or tailplane of the aircraft but when the flow over the entire aircraft is studied, all these components are in the vicinity of each other and the flow from one component will interfere with flow past another component and cause interference effects. There are several applications where we need to take into account the aerodynamics of wake due to these interference effects. The prime examples to resolve the wake downstream of an airfoil or a wing is an interaction with the fuselage and the tailplane. Another important application is the wake and flap interaction noise which is one of the main sources of noise during landing as mentioned in the experimental study by V. Hutcheson, J. Stead and E. Plassman [19]. Moreover, if the engines are installed on the rear of the aircraft as shown in Figure 1.1, the unsteady wake behavior of the flow from the wing can potentially effect the engine performance to some degree. Therefore, it is important to resolve the wake so that an optimal configuration can be obtained which will reduce drag and noise due to interference effects.

Figure 1.1: Boeing 717 (engines installed at the rear) [47]

The main objectives of this research are:

- To resolve the wake in an efficient way and keep a track of where the wake is moving as the mesh is refined. It is possible to track the wake position by tracking the wake centerline based on the velocity data (see Section 3.1).
- Compare the results of mesh adaptation with and without the wake centerline for unstructured triangular meshes using an existing metric-based anisotropic mesh adaptation scheme developed by Zuniga Vazquez [55].
- Solve the linear elasticity problem to morph the adapted mesh to match the new wake centerline in subsequent cycles of mesh adaptation.
- Use the advancing layer mesh generator in our in-housing meshing code to generate an anisotropic quad dominant advancing layer mesh and compare the results with unstructured triangular meshes with and without the wake centerline.

1.2 Thesis Outline

The overall structure of this thesis is as follows:

Chapter 2 provides background on different topics needed throughout this research. Section 2.1 briefly summarizes the finite volume solver. In Section 2.2, different mesh properties are discussed. Section 2.3 discusses the solution reconstruction as the velocity data needed to track the streamline (wake centerline) is obtained from solution reconstruction. Section 2.4 talks about the metric-based anisotropic mesh adaptation scheme. In Section 2.5, the principal operations used for mesh quality improvement by the mesh adaptation scheme are discussed. The last two sections of Chapter 2 provide an overview of streamlines and the adaptive fourth-order Runge-Kutta method used to numerically integrate the velocity data.

In Chapter 3, the overall methodology of anisotropic mesh adaptation for wakes is discussed. First, it includes wake centerline tracking using an adaptive fourth-order Runge-Kutta method from an initial isotropic unstructured mesh. The second step is to insert the wake centerline as an internal boundary into the initial unstructured mesh. Next, it talks about anisotropic mesh adaptation and re-computing the solution on the adapted mesh to obtain the new wake centerline. Then, the mesh morphing technique used to morph the old mesh to match the new wake centerline in subsequent adaptation cycles is discussed. Finally, this chapter concludes by discussing the advancing layer mesh generation scheme used to generate quad dominant meshes.

Chapter 4 of this thesis illustrates the results. The verification cases for the streamline tracking algorithm and the mesh morphing approach are included in Section 4.1. Section 4.2 contains three laminar flow test cases to demonstrate the mesh adaptation algorithm. Finally, Section 4.3 includes the streamline tracking results for the turbulent subsonic flow.

Chapter 5 ends the thesis with concluding remarks. It contains a summary of the research work, conclusions drawn based on the results and the recommendations for the future work.

Chapter 2

Background

This chapter provides details of the background material needed to do this research. In the beginning, a brief overview of finite volume solver is given. In Section 2.2, different mesh properties are discussed. Section 2.3 discusses the solution reconstruction as the velocity data needed to track the streamlines is obtained from solution reconstruction. Sections 2.4 and 2.5 include details about the metric-based anisotropic mesh adaptation and the principal operations used for mesh quality improvement respectively. The last two sections of the chapter include discussion on streamlines and the adaptive fourth-order accurate Runge-Kutta method used to numerically integrate the velocity data.

2.1 Finite Volume Solver

The equations governing the behavior of fluid flow are a set of partial differential equations (PDE's) that describe the physical conservation laws. The CFD discretization of these PDE's needs to be stable, convergent, accurate and conservative. In the finite volume method, when the flow equations are solved, a conservative discretization is satisfied automatically through the use of integral conservative laws. This means the finite volume discretization is conservative by nature and a natural choice for solving fluid related problems [8, 25, 35]. The finite volume solver begins with governing equations in conservation form, represented as:

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} + \frac{\partial H}{\partial z} = S$$

$$\frac{\partial U}{\partial t} + \nabla . \overrightarrow{F} = S, \tag{2.1}$$

with U being the solution vector, $\overrightarrow{F}$ the flux dyad and S is the source term.

Equation 2.1 is integrated over each sub-domain (known as a control volume); applying Gauss's theorem, we get

$$\int_{CV_i} \frac{\partial U}{\partial t} dV + \oint_{\partial(CV)_i} \overrightarrow{F}.\overrightarrow{n}\, dA = \int_{CV_i} S dV.$$

Assuming that the mesh is not moving, the above expression can be simplified to give

$$\frac{d\overline{U}_i}{dt} = -\frac{1}{V} \oint_{\partial(CV)_i} \overrightarrow{F}.\overrightarrow{n}\, dA + \overline{S}_i, \tag{2.2}$$

where $\overline{U}_i \equiv \frac{1}{V} \int_{CV_i} U dV$ is the average value of the solution in the control volume i and $\overline{S_i} \equiv \frac{1}{V} \int_{CV_i} S dV$ is the average source term contribution.

Equation 2.2 implies that the finite volume solution for any time varying problem can be advanced from one time level to the next with following steps:

- Compute the flux $\overrightarrow{F}$ at the surface of the control volume.
- Integrate the normal flux $\overrightarrow{F}.\overrightarrow{n}$ around the control volume boundary.
- Compute and integrate the source term S in the control volume.
- Advance the average solution value $\overline{U}$ in time.

2.2 Mesh Properties

The partial differential equations governing the behavior of fluid flow are defined over a continuum domain. In CFD, before we solve these PDE's numerically, we divide the continuous domain into a discrete domain, called the mesh. Depending on the connectivity of cells and vertices in the mesh, the mesh is classified as structured or unstructured. The type of mesh to use for CFD analysis of a certain problem depends on the discretization technique, flow physics and problem geometry. For example, the finite element and finite volume method can be used with both structured and unstructured meshes while the finite difference method can be used only with structured meshes. Apart from distinguishing the mesh based on the connectivity, another way of categorizing the mesh is by means of shape, orientation and size of cells throughout the computational domain. This classifies the mesh into categories of isotropic and anisotropic. In an isotropic mesh, the edge lengths of cells are roughly equal in all directions whereas, in an anisotropic mesh, the cells have longer edge lengths in one direction compared to the other. Each of these mesh types are discussed briefly.

2.2.1 Structured and Unstructured Meshes

In a structured mesh, there is regular connectivity between the mesh cells and vertices. All the interior cells and vertices in the mesh have a fixed number of neighbors. Figure 2.1 represents the topology for a structured mesh in which the interior cell represented by indices (i, j) has four neighbors represented by indices $(i-1, j)$, $(i+1, j)$, $(i, j+1)$ and $(i, j-1)$. These meshes usually consist of quadrilaterals in 2D and hexahedral elements in 3D.

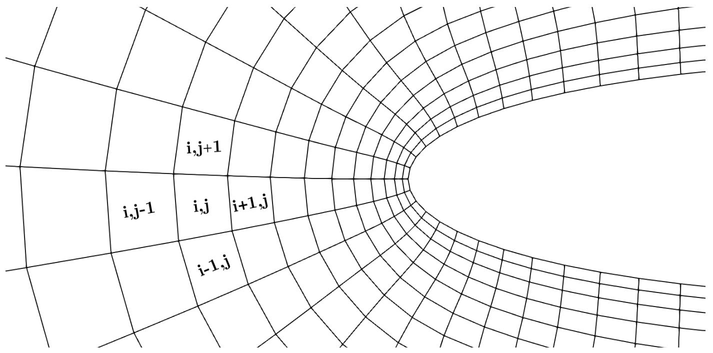


Figure 2.1: Structured mesh around leading edge of a NACA 0012 airfoil

Contrariwise, in an unstructured mesh, the mesh cells and vertices are not connected in any regular pattern. The interior cells and vertices in the mesh do not have a fixed number of neighbors. Figure 2.2 shows the topology for an unstructured mesh. Mesh resolution and grading are important factors for unstructured meshes. The mesh typically needs to finer in regions close to the solid boundary and coarser in the far-field. This is because the gradients of flow variables are high close to the solid boundary as in the boundary layer and wake region and so finer mesh resolution is required in those regions of the domain for better solution accuracy. The coarser mesh in the far-field where changes are small helps to reduce the overall computational cost. Unstructured meshes usually consist of triangles or quadrilaterals in 2D and tetrahedra, prisms or pyramids in 3D [23, 54].

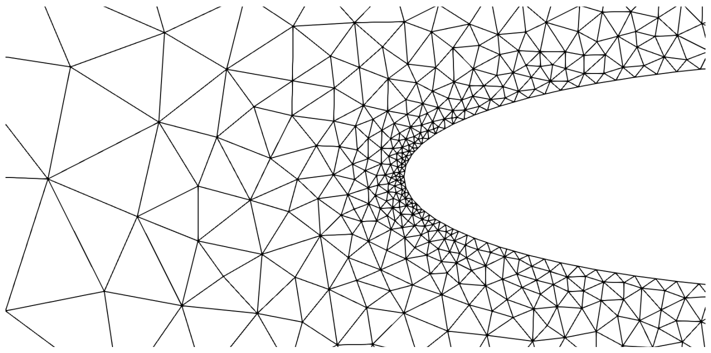


Figure 2.2: Unstructured mesh around leading edge of NACA 0012 airfoil

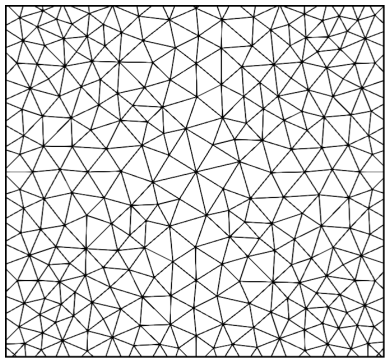
For structured meshes, there is no need to store the data for each mesh vertex explicitly since the neighboring vertices in the physical space are also neighboring elements in the connectivity matrix. This reduces the memory usage which is an advantage from the computational point of view. For unstructured meshes, there is more memory usage due to the irregular topology of unstructured mesh: the connectivity of each vertex in a mesh to other vertices needs to be stored explicitly [27, 52].

In structured meshes, due to the fixed topology, less CPU time is required by the finite volume solver for evaluation of numerical fluxes, because the solution needed for flux evaluation can be computed by one-dimensional interpolation. For unstructured meshes, accurate flux evaluation requires polynomial reconstruction of solution (See Section 2.3) over each control volume which increases the CPU time.

However, many industrial level problems involve complex geometries and generation of structured meshes around complex geometries is a major challenge. Although less CPU time is required to compute the numerical flux, the drawback of fully structured meshes is that we don't have the ability to put all the resolution where it is needed and we can end up with excess resolution in some places. More specifically, we can't independently control cell spacing in different parts of the domain and excess resolution increases CPU time. Also, generating a structured mesh around complex shapes can require time-consuming human interventions such as splitting the domain into multiple blocks depending on the complexity of the geometry. Such factors can offset the CPU time advantage gained in the evaluation of numerical fluxes by the solver. Unstructured meshes are much easier to generate around complex geometries and usually a preferred choice for complex configurations. Although the polynomial reconstruction of the solution over each control volume increases CPU time, unstructured meshes have the ability to capture the flow features around complex geometries with fewer cells. This leads to a considerable reduction in overall CPU time. Also, it is easier to do adaptive mesh refinement since cell regularity is not an issue [32, 33].

2.2.2 Isotropic and Anisotropic Meshes

The accuracy of numerical solutions is greatly influenced by the characteristics such as the shape, orientation and size of mesh elements. Figure 2.3a shows an isotropic mesh fragment with triangular elements in which the edge lengths of cells are approximately equal. In other words, triangular elements are roughly equilateral triangles. Contrariwise, an anisotropic mesh has cells with higher aspect ratio as depicted in Figure 2.3b.


(a) Isotropic mesh

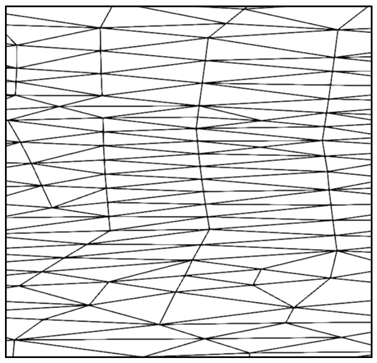

(b) Anisotropic mesh

Figure 2.3: Isotropic and anisotropic mesh fragments

The choice of using an isotropic or an anisotropic mesh depends on the nature of the physical processes occurring in the problem. If the physical changes are isotropic, it is desirable to use an isotropic mesh with a cell size that equidistributes the error. However, the flow field in aerodynamics applications is usually accompanied by phenomena such as turbulence, boundary layers, wakes and shocks that have anisotropic physical behavior. As mentioned earlier, resolving these flow features using isotropic elements is computationally inefficient [14]. Moreover, isotropic meshes can provide only the proper mesh resolution while anisotropic mesh adaptation gives both mesh resolution and proper cell alignment with the solution to capture anisotropic flow features. Therefore, such complex flow features can be resolved more efficiently using anisotropic cells.

2.3 Solution Reconstruction

The accuracy of the finite volume solution is highly dependent on the accuracy of flux integrals which in turn requires accurate approximation of the unknown variables in the control volume. In order to approximate the unknown variables, represented by ϕ, in the flow field, the flow solver aims to replace the control volume average $\overline{\phi}_i$ for each interior control volume in the mesh with a Taylor series about a control volume reference point (x_i, y_i) [39]:

$$\begin{aligned}\overline{\phi}_i = \phi_i^R(x,y) = \phi\big|_i + \left.\frac{\partial \phi}{\partial x}\right|_i (x-x_i) + \left.\frac{\partial \phi}{\partial y}\right|_i (y-y_i) \\ + \left.\frac{\partial^2 \phi}{\partial x^2}\right|_i \frac{(x-x_i)^2}{2} + \left.\frac{\partial^2 \phi}{\partial x \partial y}\right|_i (x-x_i)(y-y_i) \\ + \left.\frac{\partial^2 \phi}{\partial y^2}\right|_i \frac{(y-y_i)^2}{2} + ..., \end{aligned} \tag{2.3}$$

where

$\overline{\phi}_i$ is the average value of the solution in control volume i

$\phi_i^R(x,y)$ is the value of reconstructed solution at a point (x,y) for control volume i

$\frac{\partial^{k+1}\phi_i}{\partial x^k \partial y^i}$ are the derivatives of the reconstructed solution for control volume i

The values ϕ_i and $\frac{\partial^{k+1}\phi_i}{\partial x^k \partial y^i}$ correspond to coefficients of Taylor polynomials. The accuracy of the solution is one order higher than the polynomial degree. These coefficients are chosen so that we conserve the mean value of the solution in the control volume i, that is, we must satisfy the following criterion:

$$\frac{1}{A_i}\int_{V_i} \phi_i^R dA = \overline{\phi}_i \tag{2.4}$$

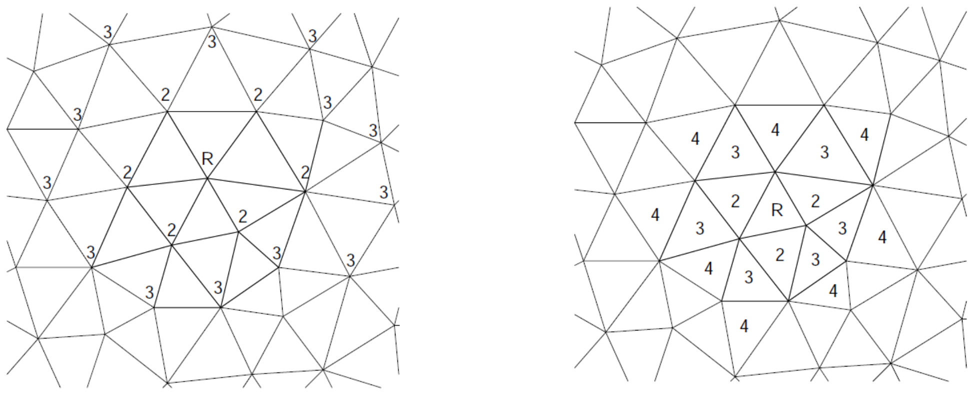
Combining Equations 2.3 and 2.4 leads to:

$$\begin{aligned}\overline{\phi}_i = \frac{1}{A_i}\int_{V_i} \phi_i^R dA = \phi\Big|_i + \frac{\partial \phi}{\partial x}\Big|_i \overline{x}_i + \frac{\partial \phi}{\partial y}\Big|_i \overline{y}_i + \frac{\partial^2 \phi}{\partial x^2}\Big|_i \frac{\overline{x_i^2}}{2} \\ + \frac{\partial^2 \phi}{\partial x \partial y}\Big|_i \overline{xy}_i + \frac{\partial^2 \phi}{\partial y^2}\Big|_i \frac{\overline{y_i^2}}{2} + ...,\end{aligned} \tag{2.5}$$

with $\overline{x^n y^m}_i$ being the moments of area such that:

$$\overline{x^n y^m}_i = \frac{1}{A_i}\int_{V_i} (x - x_i)^n (y - y_i)^m dA. \tag{2.6}$$

The accuracy of a least-squares reconstruction scheme for smooth functions can be said, equivalently, to be k-exact or $(k+1)$ order accurate and requires us to expand the modified Taylor series, represented by Equation 2.5, up to the $k-th$ derivatives. While evaluating the derivatives, we try to minimize errors in the average value of ϕ_i^R (reconstructed solution) for a nearby set of control volumes called the stencil $\{V_j\}_i$.



(a) Vertex-centered control volume

(b) Cell-centered control volume

Figure 2.4: Reconstruction stencils [39]

In the reconstruction stencil, we require at least as many control volumes as derivative terms that we need to compute. However, practically we exceed this minimum number of control volumes for each order of accuracy, thus increasing the robustness of the least-squares reconstruction scheme. For example, for second-order accuracy, we have three neighboring control volumes and for third-order accuracy, we have nine neighboring control volumes. The approach to adding control volumes for both vertex-centered and cell-centered simulations is to add all the control volumes at the same topological distance to the reconstruction stencil at once. Figure 2.4 above shows the control volumes for the vertex-centered and cell-centered cases with control volume labeled R representing the control volume to be reconstructed and the numbers represent the order of accuracy at which each neighboring control volume is added.

The average value of ϕ_i^R (reconstructed function) for each control volume in the stencil will be:

$$\begin{aligned}\frac{1}{A_j}\int_{V_j}\phi_i^R(\overrightarrow{x}-\overrightarrow{x}_i)dA = \phi\big|_i &+ \frac{\partial\phi}{\partial x}\bigg|_i \frac{1}{A_j}\int_{V_j}(x-x_i)dA \\ &+\frac{\partial\phi}{\partial y}\bigg|_i \frac{1}{A_j}\int_{V_j}(y-y_i)dA \\ &+\frac{\partial^2\phi}{\partial x^2}\bigg|_i \frac{1}{2A_j}\int_{V_j}(x-x_i)^2dA \\ &+\frac{\partial^2\phi}{\partial x\partial y}\bigg|_i \frac{1}{A_j}\int_{V_j}(x-x_i)(y-y_i)dA \\ &+\frac{\partial^2\phi}{\partial y^2}\bigg|_i \frac{1}{2A_j}\int_{V_j}(y-y_i)^2dA + ... \end{aligned} \tag{2.7}$$

In Equation 2.7, $x-x_i$ and $y-y_i$ are replaced with $(x-x_j)+(x_j-x_i)$ and $(y-y_j)+(y_j-y_i)$ to avoid computing moments for each control volume of $\{V_j\}_i$ about the reference point of control volume i. Equation 2.7 then becomes

$$\begin{aligned}\frac{1}{A_j}\int_{V_j}\phi_i^R(\overrightarrow{x}-\overrightarrow{x}_i)dA &= \phi_i+\left.\frac{\partial\phi}{\partial x}\right|_i(\overline{x}_j+(x_j-x_i))+\left.\frac{\partial\phi}{\partial y}\right|_i\Big(\overline{y}_j+(y_j-y_i)\Big)\\ &+\left.\frac{\partial^2\phi}{\partial x^2}\right|_i\frac{\overline{x^2}_j+2\overline{x}_j\,(x_j-x_i)+(x_j-x_i)^2}{2}\\ &+\left.\frac{\partial^2\phi}{\partial x\partial y}\right|_i\Big(\overline{xy}_j+\overline{x}_j\,(y_j-y_i)+(x_j-x_i)\,\overline{y}_j+(x_j-x_i)\,(y_j-y_i)\Big)\\ &+\left.\frac{\partial^2\phi}{\partial y^2}\right|_i\frac{\overline{y^2}_j+2\overline{y}_j\,(y_j-y_i)+(y_j-y_i)^2}{2}+...\end{aligned}\tag{2.8}$$

The geometric terms in above equation are mesh dependent and have the following form

$$\begin{aligned}\widehat{x^ny^m}_{ij} &\equiv \frac{1}{A_j}\int_{V_j}((x-x_j)+(x_j-x_i))^n\cdot((y-y_j)+(y_j-y_i))^m\,dA\\ &=\sum_{l=0}^{n}\sum_{k=0}^{m}\frac{n!}{l!\,(n-l)!}\frac{m!}{k!\,(m-k)!}\,(x_j-x_i)^k\cdot(y_j-y_i)^l\cdot\overline{x^{n-k}y^{m-l}}_j\end{aligned}$$

Equation 2.8 then becomes

$$\begin{aligned}\frac{1}{A_j}\int_{V_j}\phi_i^R(\overrightarrow{x}-\overrightarrow{x}_i)dA &= \phi|_i+\left.\frac{\partial\phi}{\partial x}\right|_i\widehat{x}_{ij}+\left.\frac{\partial\phi}{\partial y}\right|_i\widehat{y}_{ij}\\ &+\left.\frac{\partial^2\phi}{\partial x^2}\right|_i\frac{\widehat{x^2}_{ij}}{2}+\left.\frac{\partial^2\phi}{\partial x\,\partial y}\right|_i\widehat{xy}_{ij}+\left.\frac{\partial^2\phi}{\partial y^2}\right|_i\frac{\widehat{y^2}_{ij}}{2}+\cdots\end{aligned}\tag{2.9}$$

Equation 2.9 gives the average value of the reconstructed solution $\phi_i^R(\overrightarrow{x}-\overrightarrow{x}_i)$ for control volume j. The difference between the actual control volume average and the average value of the reconstructed solution can be easily accessed. The derivatives at $\overrightarrow{x}_i$ are chosen to minimize the magnitude of error in predicting the control volume average values in a least-squares sense [39].

The resulting least-squares system to compute the derivatives is

$$\left[\begin{array}{ccccccc} 1 & \overline{x}_i & \overline{y}_i & \overline{x^2}_i & \overline{xy}_i & \overline{y^2}_i & \cdots \\ \hline w_{i1} & w_{i1}\widehat{x}_{i1} & w_{i1}\widehat{y}_{i1} & w_{i1}\widehat{x^2}_{i1} & w_{i1}\widehat{xy}_{i1} & w_{i1}\widehat{y^2}_{i1} & \cdots \\ w_{i2} & w_{i2}\widehat{x}_{i2} & w_{i2}\widehat{y}_{i2} & w_{i2}\widehat{x^2}_{i2} & w_{i2}\widehat{xy}_{i2} & w_{i2}\widehat{y^2}_{i2} & \cdots \\ w_{i3} & w_{i3}\widehat{x}_{i3} & w_{i3}\widehat{y}_{i3} & w_{i3}\widehat{x^2}_{i3} & w_{i3}\widehat{xy}_{i3} & w_{i3}\widehat{y^2}_{i3} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \\ w_{iN} & w_{iN}\widehat{x}_{iN} & w_{iN}\widehat{y}_{iN} & w_{iN}\widehat{x^2}_{iN} & w_{iN}\widehat{xy}_{iN} & w_{iN}\widehat{y^2}_{iN} & \cdots \end{array}\right] \begin{pmatrix} \phi \\ \frac{\partial \phi}{\partial x} \\ \frac{\partial \phi}{\partial y} \\ \frac{1}{2}\frac{\partial^2 \phi}{\partial x^2} \\ \frac{\partial^2 \phi}{\partial x\,\partial y} \\ \frac{1}{2}\frac{\partial^2 \phi}{\partial y^2} \\ \vdots \end{pmatrix}_i = \left(\begin{array}{c} \bar{\phi}_i \\ \hline w_{i1}\bar{\phi}_1 \\ w_{i2}\bar{\phi}_2 \\ w_{i3}\bar{\phi}_3 \\ \vdots \\ w_{iN}\bar{\phi}_N \end{array}\right)$$

In the least-squares system, w_{ij} represents the geometric weights that are based on the distance between control volume reference points. These geometric weights can be used to specify the relative importance of neighboring control volumes. N is the number of nearby control volumes in the reconstruction stencil. This constrained least-squares system is converted into an unconstrained least-squares system by eliminating the first row representing the mean constraints by using Gauss elimination. The remaining unconstrained least-squares problem is then solved for each control volume using the singular value decomposition (SVD) method [15]. This is then followed by flux evaluation and flux integration, the details of which can be found in reference [3, 39].

2.4 Metric-Based Anisotropic Mesh Adaptation

As stated in Chapter 1, the purpose of generating anisotropic meshes is to reduce the computational cost for a given level of solution accuracy. Anisotropic mesh cells help to achieve this goal by capturing the solution features with fewer cells. Moreover, unlike isotropic mesh adaptation, anisotropic adaptation also provides good cell alignment in addition to the mesh resolution in areas of interest.

A robust anisotropic mesh adaptation scheme demands good error estimation and high quality anisotropic cells. Many researchers [4, 6, 12, 30] have illustrated that selectively refining/coarsening the mesh cells and producing highly anisotropic meshes is effective. Different methods exist for error estimation. This thesis uses the anisotropic mesh error estimate implemented by Pagnutti [41]. This scheme makes use of local reconstruction error and gives an error measure based on the overall solution vector rather than any specific functional as in adjoint error estimation. Zuniga Vazquez's [55] anisotropic mesh adaptation scheme was used in this research. The robustness of the mesh adaptation scheme is further demonstrated with different test cases in this thesis.

The concept behind metric-based adaptation is to produce a unit mesh in Riemannian metric space. The error estimate gives the desired anisotropy which is stored in a metric tensor at each vertex in the computational domain. The desired anisotropy is then communicated to the mesh adaptation code to produce an anisotropic mesh. For 2D problems, the metric is a 2×2 symmetric positive definite tensor. Geometrically, the metric at a point tells how distance and angles are measured as seen from that point. If we have a metric tensor M_p, then the deformation tensor L_p can be defined as:

$$M_p = L_p^T L_p \qquad \text{and} \qquad det\,(L_p) > 0 \tag{2.10}$$

with L_p being an invertible matrix that maps physical space into metric space as depicted in Figure 2.5 and the distances and angles are measured in the Euclidean way as seen by point p'.

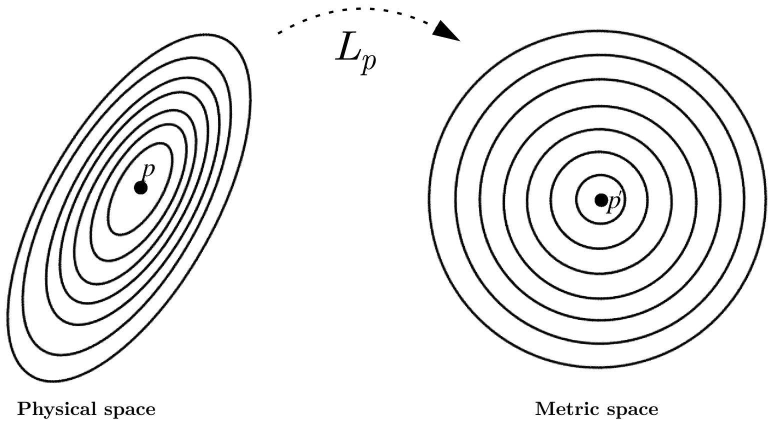


Figure 2.5: Deformation of tensor L_p

The ellipses around point p' in Figure 2.5 are equidistant from each other in metric space and define the anisotropy using three components: radius r_1, radius r_2 and angle θ as shown in Figure 2.6. Then, the metric is represented by these three components as:

$$M = R\Lambda R^T = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} 1/r_1^2 & 0 \\ 0 & 1/r_2^2 \end{pmatrix} \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \tag{2.11}$$

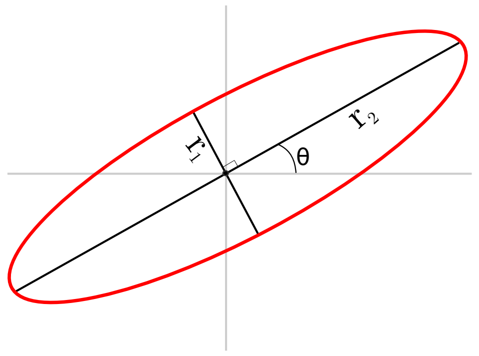


Figure 2.6: Anisotropy components for a certain point

A metric is a distance function $d : X \times X \to \mathbb{R}$ that defines the positive distance measure between any two distinct points in a vector space X and fulfills the following conditions:

$$\begin{aligned} d(x,y) &> 0 \iff x \neq y && \text{non-negativity} \\ d(x,y) &= d(y,x) && \text{symmetry} \\ d(x,z) &\leq d(x,y) + d(y,z) && \text{triangle inequality} \end{aligned} \tag{2.12}$$

In the case of a uniform isotropic mesh, the metric is the standard Cartesian distance $d_{\text{std}}(x,y) = \sqrt{(x-y)^T(x-y)}$. The metric for anisotropic adaptation $d_{\text{ani}}(x,y)$ can be obtained by the invertible linear transformation L of $d_{\text{std}}(x,y)$.

$$\begin{aligned}
d_{\text{std}}(x,y) &= \sqrt{(x-y)^T(x-y)} \\
d_{\text{ani}}(x,y) &= \sqrt{(L(x-y))^T\,(L(x-y))} \\
d_{\text{ani}}(x,y) &= \sqrt{(x-y)^T\,(L^TL)\,(x-y)} \\
d_{\text{ani}}(x,y) &= \sqrt{(x-y)^TM(x-y)} \qquad (2.13)
\end{aligned}$$

This illustrates that the metric $d_{\text{ani}}(x,y)$ can be represented by a positive definite matrix $M = L^TL$. Further details on the complete metric computation and its suitability to generate meshes can be found in references [41, 49].

2.4.1 Aspect ratio and orientation from metric

The metric is given by:

$$M = \begin{bmatrix} M_{xx} & M_{xy} \\ M_{xy} & M_{yy} \end{bmatrix}$$

From Mohr's circle, we have:

$$M_{max} = \frac{M_{xx} + M_{yy} + \sqrt{(M_{xx} - M_{yy})^2 + 4M_{xy}^2}}{2}$$

$$M_{min} = \frac{M_{xx} + M_{yy} - \sqrt{(M_{xx} - M_{yy})^2 + 4M_{xy}^2}}{2}$$

The orientation is given by:

$$\theta = \frac{1}{2}arctan\left(\frac{2M_{xy}}{M_{xx} - M_{yy}}\right) \qquad (2.14)$$

The aspect ratio is given by:

$$AR = \sqrt{\frac{M_{max}}{M_{min}}} \qquad (2.15)$$

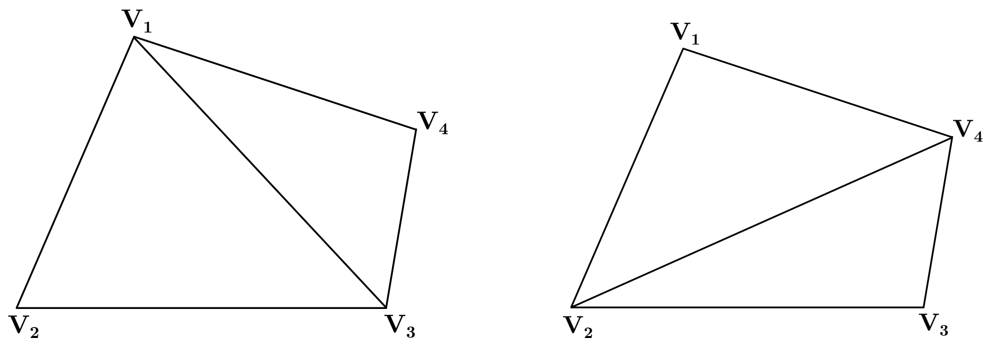
2.5 Operations for Mesh Quality Improvement

The anisotropic mesh adaptation scheme used in this research uses four principal operations for mesh quality improvement. These include face swapping in metric space, vertex insertion based on quality and maximum metric length, vertex removal (coarsening) based on the minimum metric length and vertex movement (smoothing) based on quality. The first three operations are implemented by GRUMMP[1] [38], our in-house meshing code, while vertex movement is done using Mesquite [24].

2.5.1 Face Swapping

Face swapping changes the connectivity of vertices in the mesh but the number of mesh vertices and cells remain unchanged. The new connectivity is decided such that the maximum angle in the metric space is minimized and new triangles with better quality are obtained. Figure 2.7 shows the triangles before and after face swapping. The necessary condition for face swapping is:

$$Quality\,(V_1V_2V_3) + Quality\,(V_1V_3V_4) < Quality\,(V_1V_2V_4) + Quality\,(V_2V_3V_4)$$



(a) Initial configuration (b) Modified configuration

Figure 2.7: Face swapping in metric space

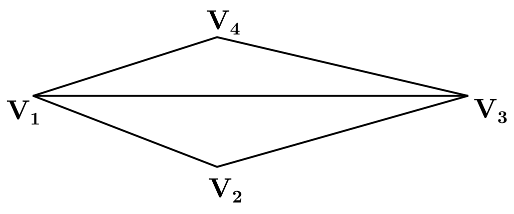
[1]Generation and Refinement of Unstructured Mixed-Element Meshes in Parallel

2.5.2 Vertex Insertion

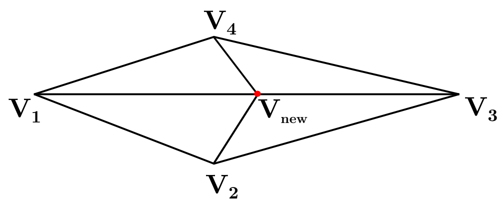
The vertex insertion or edge splitting is done based on two criteria and an insertion queue is used for this operation. The first criterion is based on quality and so the vertex is inserted onto the edge if it will improve the quality and higher priority in the insertion queue is given to the cells with the worst quality. The second criterion is related to maximum metric length and so edges longer than a certain length (based on the metric) are split and higher priority is given to the edge with longest metric length. However, to prevent the creation of extra edges, an edge is not placed in the insertion queue if it will result in an edge smaller than the desired minimum length in the metric space. This means that the desired length is such that:

$$l_M^{min} \leq l_M \leq l_M^{max}$$

The vertex insertion for an interior and boundary edge is shown in Figures 2.8 and 2.9 respectively. This operation increases mesh vertices by one and cells by two for interior edges while vertices and cells both by one for boundary edges. The same criterion is used for both the boundary and interior edges. After vertex insertion, face swapping is done as required to improve the mesh quality.

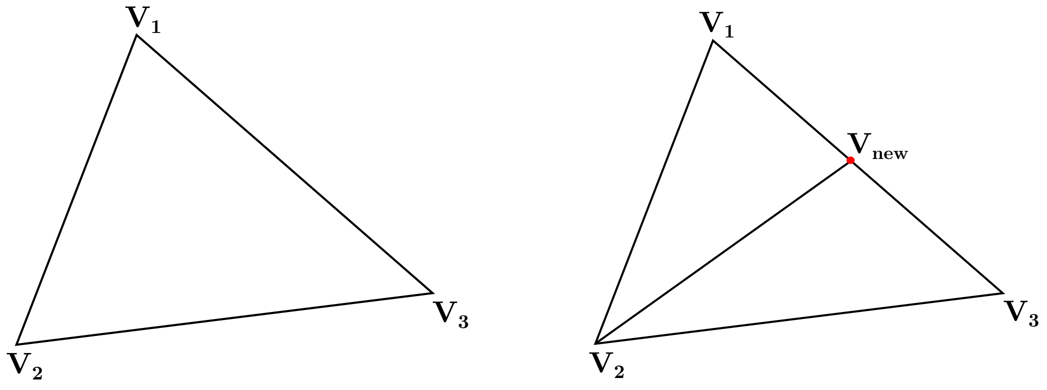


(a) Initial configuration



(b) Modified configuration

Figure 2.8: Vertex insertion for an interior edge



(a) Initial configuration (b) Modified configuration

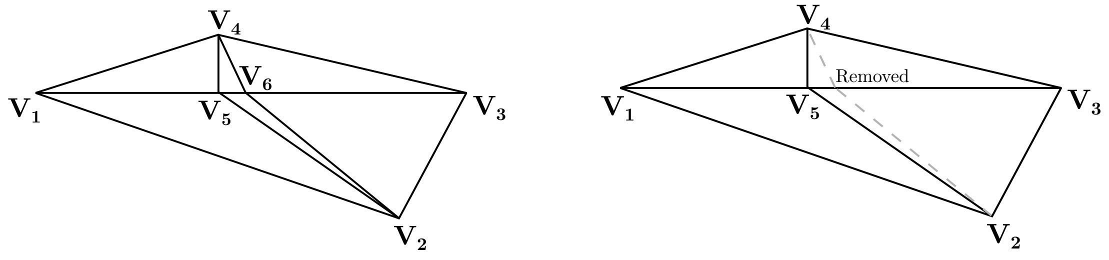
Figure 2.9: Vertex insertion for a boundary edge

2.5.3 Vertex Removal

The third mesh modification technique is vertex removal which coarsens the mesh and thereby helps to reduce the computational cost. We get rid of edges that have a small metric length. Again a priority queue is used and higher priority is given to the edge with the shortest metric length. This means that the desired length is such that:

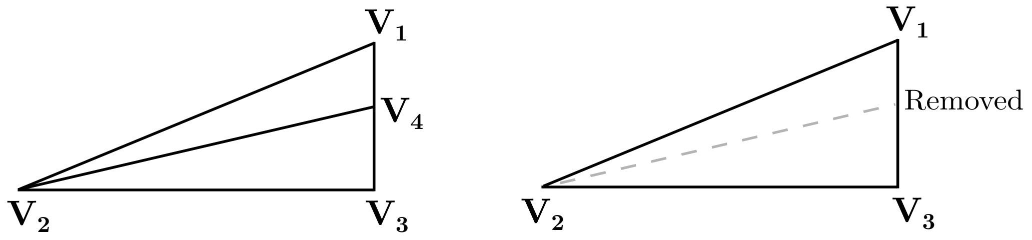
$$l_M \geq l_M^{min}$$

This operation is again followed by face swapping as required to improve mesh quality. Figures 2.10 and 2.11 shows the process of vertex removal for an interior and boundary edge respectively.



(a) Initial configuration (b) Modified configuration

Figure 2.10: Vertex removal for an interior edge



(a) Initial configuration (b) Modified configuration

Figure 2.11: Vertex removal for a boundary edge

2.5.4 Vertex Movement

The vertex movement operation is based on the Target-Matrix Optimization Paradigm (TMOP) as used in Mesquite [7, 24]. The mesh vertices are moved or repositioned in such a way that an optimal mesh of superior quality is obtained based on the target matrix. The target matrix depends on the shape of the target element. In the case of anisotropic meshes, the target element should be a right triangle to produce quasi-structured meshes [49]. The shape (aspect ratio and orientation) of each cell in an anisotropic mesh is determined from the metric and so the metric can also be used to calculate the target matrix. Since the metric information is stored at each vertex of the mesh element, an average metric is used to calculate the target matrix for each cell. More details on TMOP and average metric calculation can be found in Sharbatdar's thesis [49].

Further details on above mesh quality operations can be found in [41, 49, 55].

2.6 Streamlines

A streamline can be defined as a path traced out by a massless particle propagating with the fluid velocity. The streamline is tangent everywhere to the local velocity field or instantaneous velocity. Therefore, we need to consider one instant in time; for unsteady flow, these streamlines can look very different at each instant in time as they represent the direction of the flow (there is no flow across streamlines) [1]. This is what makes streamlines immensely useful for investigating different types of flow. The velocity fields evaluated numerically can be called CFD velocity fields and these discrete CFD velocity fields, defined on a discrete computational domain, are actually an approximation to the exact mass conservative velocity fields [26]. Figure 2.12 shows streamlines around the NACA 0012 airfoil at $Re = 5000$, $Ma = 0.5$ and $\alpha = 0°$.

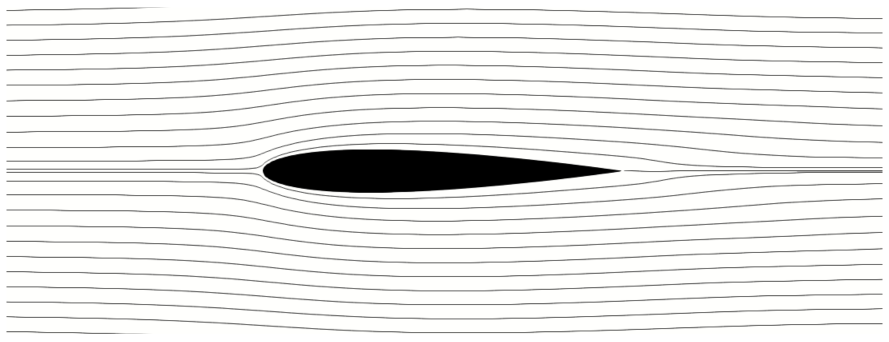


Figure 2.12: Streamlines around NACA 0012 airfoil at $Re = 5000$, $Ma = 0.5$ and $\alpha = 0^{\circ}$

It is possible to obtain a streamline equation from the velocity field by using velocity vectors in each direction in the flow. For a two-dimensional velocity field, this can be expressed in differential form as:

$$\frac{dx}{dt} = u\left(x, y\right) \tag{2.16}$$

$$\frac{dy}{dt} = v\left(x, y\right) \tag{2.17}$$

Equations 2.16 and 2.17 then need to be solved simultaneously. We can integrate them numerically to obtain $x\left(t\right)$ and $y\left(t\right)$ along the streamline.

2.7 Runge-Kutta Method and Adaptive Stepsize Control

The velocity data obtained from solution reconstruction is integrated numerically to obtain the streamline. The Runge-Kutta method is one of the classical methods used to numerically integrate ordinary differential equations (ODE's). Several versions of the Runge-Kutta method are available but in this thesis, we use the fourth-order accurate Runge-Kutta method

(RK4) to numerically integrate the ODE representing the velocity field. For an initial value problem such as:

$$\dot{x} = f\left(x, t\right) \qquad x\left(t_0\right) = x_0 \tag{2.18}$$

RK4 can be expressed as:

$$x_{i+1} = x_i + \frac{h}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) + O\left(h^5\right) \tag{2.19}$$

$$t_{i+1} = t_i + h$$

where:

$$\begin{aligned} k_1 &= f\left(x_i, t_i\right) \\ k_2 &= f\left(x_i + \frac{k_1 h}{2}, t_i + \frac{h}{2}\right) \\ k_3 &= f\left(x_i + \frac{k_2 h}{2}, t_i + \frac{h}{2},\right) \\ k_4 &= f\left(x_i + k_3 h, t_i + h\right) \end{aligned}$$

and x_{i+1} is the numerical approximation of $x\left(t_{i+1}\right)$ [50].

For each step, Equation 2.19 requires four computations of the right-hand side. These computations are the values of the derivative at a point: k_1 is the value at the starting point of the interval, k_2 and k_3 are the values at the first and second trial midpoints respectively and k_4 at the trial endpoint. The final function in Equation 2.19 is then obtained by taking the average of the four derivatives with higher weight given to the derivatives at the trial midpoints.

An efficient differential equation integrator should have some stepsize control based on an error estimate. Adaptive stepsize control can be implemented for the RK4 scheme based on an estimate of the local truncation error. In this case, each step is taken twice, once as a full step and then as two half steps and the difference between the two numerical estimates is the truncation error. Figure 2.13 illustrates the adaptive stepsize control for RK4, with filled squares representing the points at which the derivatives are evaluated. The big step and two small steps procedure share the starting point represented by the empty square for the small step so no extra elevation is required at that point and the total number of evaluations for two small steps will be 11. Since reducing the step to half also drops the error, the 11 evaluations per two small steps should be compared with 8 and not 4 evaluations of the big step. Therefore, the factor for the additional cost is 1.375.

If we denote the exact solution by $x\left(t+2h\right)$ while advancing from t to $t+2h$ and the numerical estimates by x_1 and x_2, then [44]:

$$x\left(t+2h\right) = x_1 + \left(2h\right)^5 \phi \ + O\left(h^6\right)$$

$$x\left(t+2h\right) = x_2 + 2\left(h\right)^5 \phi \ + O\left(h^6\right)$$

$$\text{Truncation error} \equiv \ x_2 - x_1$$

If the estimated error is above the user-specified tolerance for the truncation error, then the step is repeated and stepsize is cut in half until the tolerance criteria is satisfied. Similarly, if the error is too small compared to the tolerance, then the step is repeated and the stepsize is doubled to save the computational cost. This process provides optimal stepsize automatically and the user does not need to spend time on finding the optimal stepsize each time.

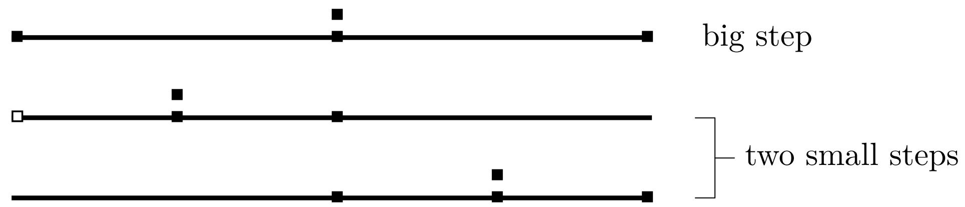


Figure 2.13: Adaptive stepsize control for RK4 [44]

Chapter 3

Methodology

The approach for anisotropic mesh adaptation is divided into several steps. This chapter gives an overview of each step and summarizes how our mesh adaptation loop works. In the end, it discusses the advancing layer mesh generation which is used as an alternate technique to generate an anisotropic quad dominant mesh.

3.1 Wake Centerline Tracking

The algorithm to track the streamlines is implemented in our in-house solver code, ANSLib [40]. For 2D airfoil flows, we can find the wake centerline by tracking a streamline from the trailing edge. A massless particle propagating with the fluid velocity is released in the flow field and its location tracked in time to see where the streamline goes, as the streamline is tangent to the velocity vector. Note that we do not integrate the flow in time but rather the motion of particle within the steady flow. The velocity data is obtained from the solution reconstruction using the vertex-centered finite volume solver. For linear reconstruction, the ODE for the streamline location is:

$$\frac{d}{dt}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u_0 + a_1\left(x - x_i\right) + a_2\left(y - y_i\right) \\ v_0 + b_1\left(x - x_i\right) + b_2\left(y - y_i\right) \end{pmatrix} \tag{3.1}$$

In the above equation, u_0 and v_0 are the x and y components of velocity respectively at the initial point (x_0, y_0) where the streamline enters the control volume. Moreover, a_1, a_2, b_1, and b_2 are the derivatives of the reconstructed solution at any point (x_i, y_i) in the control volume i.

We integrate this system of ODE's in time using an adaptive fourth-order Runge-Kutta method. A geometric search tree is used to identify which control volume the particle is in based on its position at a given time. The adaptive time step scheme for numerical integration ensures that the local truncation error remains within a user-specified tolerance (see Section 2.7).

Furthermore, we get a steady separation in the wake at different flow conditions. Figure 3.1 shows the streamlines for a separated flow. In case of flow separation, numerically integrating Equation 3.1 forward in time is not sufficient. To track the wake centerline, we need to march downstream from the trailing edge and so $x\,(t+\Delta t)$ should be always higher than the previous value $x\,(t)$. Simply doing forward integration in time from the trailing edge can sometimes take us inside in the airfoil (for example at Reynolds number $Re = 5000$, Mach number $Ma = 0.5$ and angle of attack $\alpha = 0^{\circ}$) rather than downstream from the trailing edge. In such a case, we need to start with backward integration in time to march in the right direction. Therefore, in the streamline tracking algorithm, we always start with backward integration in time and if it takes us upstream (the next value of $x\,(t)$ will be less than 1), we exit the backward integration cycle immediately at the trailing edge and do only forward integration in time to move downstream. However, if backward integration takes us downstream from the trailing edge, then there is a saddle point along the wake centerline and the backward integration cycle is terminated once we hit the saddle point. An offset is added to escape the separation bubble, followed by forward integration in time to move downstream from the separated region. The offset that needs to be added is usually between 1% and 3% of the chord length but can be slightly higher — 5% to 6% of chord length — when separation is large at higher angles of attack. Moreover, one exception is that there can be a saddle point in the flow-field along the wake centerline even when forward integration in time initially takes us downstream from the trailing edge. In that case (for example at $Re = 5000$, $Ma = 0.5$ and $\alpha = 10^{\circ}$) two cycles of forward integration in time are required. We exit the first cycle of forward integration in time as soon as we hit the saddle point and the value of $x\,(t)$ starts to decreases after that point. An offset is added to escape the separation bubble and we then do second forward integration in time to move downstream again.

Because streamline integration does not exactly hit the saddle point, we identify the saddle point as the first point where $|v| > |u|$ and the value of $x(t)$ obtained from numerical integration is higher than the previous value of $x(t)$. While inexact, this is a very close approximation to the saddle point location. As illustrated by Figure 3.1, for separated flows, the wake centerline can be tracked in several steps: (1) backward or forward integration in time from the trailing edge, (2) find the saddle point along the wake centerline, and (3) add an offset from the saddle point to escape the separation bubble and do forward integration in time to continue downstream.

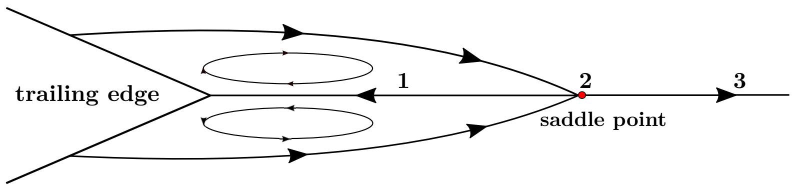


Figure 3.1: Separated flow

3.2 Wake Centerline Insertion into Unstructured Mesh

The obtained streamline is inserted as an internal boundary into the existing isotropic unstructured mesh, using the surface insertion algorithm implemented in GRUMMP by Zaide and Ollivier-Gooch [60]. Inserting the streamline as an internal boundary helps to deal with physical behavior in the wake and enables the pre-existing mesh to adapt to that behavior. The surface insertion approach starts with two initial inputs: an initial unstructured mesh and information about the geometry to be inserted. This approach has no requirement on the initial mesh and rather than moving the existing vertices onto the surface, new vertex locations on the boundary curve are computed and points are inserted to match the surface. The algorithm samples the boundary curve on the mesh, using the length scale of the mesh to determine edge lengths. Vertices in the original mesh near the curve are removed, the new vertices on the sampled curve are inserted into the mesh and surface recovery is done. The surface insertion technique gives freedom to work with an arbitrary initial mesh but the main point of this approach is to recover the surface rather than moving points while not changing the mesh at all more than a cell size or two from the wake surface. Once the surface is recovered after insertion, the mesh quality is improved with swapping, smoothing and refinement. Further details on the surface insertion technique can be found in the paper by Zaide and Ollivier-Gooch [60].

3.3 Anisotropic Mesh Adaptation

After inserting the wake centerline into the existing unstructured mesh, the next step is to do metric-based anisotropic mesh adaptation. As explained earlier, our metric is based on solution approximation error, and the error estimation is done by interpolation of the solution between known values. For a second-order accurate solution, the solution approximation error

$$\frac{1}{2} f_{xx}\left(x, y\right) x^2 + f_{xy}\left(x, y\right) xy + \frac{1}{2} f_{yy}\left(x, y\right) y^2,$$

can be related to the Hessian of the solution

$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix}$$

A metric function is chosen such that it gives a good approximation of the solution error. It is approximated by doing a Fourier transform around a unit circle of the maximum error in any solution component and the Fourier coefficients are obtained with the help of numerical integration. The combinations of Fourier coefficients are then used as metric terms [42, 41].

As explained in Section 2.5, the anisotropic mesh adaptation scheme uses four operations for mesh quality improvement in the metric space but it preserves the wake centerline. The adapted mesh is used to compute the flow solution again, using the second-order accurate vertex-centered finite volume scheme and the velocity field is integrated to obtain the new wake centerline.

3.4 Mesh Morphing

In the past, researchers [21, 58] have used the linear elasticity method to deform the mesh in order to take into account the effect of curvature especially for higher-order solvers to accurately represent the boundary faces. Jalali and Ollivier-Gooch [21] implemented this approach in our in-house code for curving the boundary faces and solved the linear elasticity problem using the finite element discretization. In this thesis, we extend their approach further and take into account linear elements and change the boundary condition to displace the wake centerline.

In the second and subsequent mesh adaptation cycles, we begin by morphing the mesh to match the new wake centerline. We project the mesh points along the old wake centerline onto the new wake centerline and solve a linear elasticity problem with displacements between the two wake centerlines as constraints to morph the rest of the mesh. After this, we apply metric-based adaptation to the morphed mesh. Because the morphing process preserves existing adaptation in the wake region, the metric adaptation is more efficient. The linear elasticity equation is given by:

$$\begin{aligned}
\frac{\partial}{\partial x}\left(d_{11}\frac{\partial \delta_x}{\partial x} + d_{12}\frac{\partial \delta_y}{\partial y}\right) + \frac{\partial}{\partial y}\left(d_{33}\left(\frac{\partial \delta_x}{\partial y} + \frac{\partial \delta_y}{\partial x}\right)\right) &= 0 \\
\frac{\partial}{\partial x}\left(d_{33}\left(\frac{\partial \delta_x}{\partial x} + \frac{\partial \delta_y}{\partial y}\right)\right) + \frac{\partial}{\partial y}\left(d_{21}\frac{\partial \delta_x}{\partial y} + d_{22}\frac{\partial \delta_y}{\partial x}\right) &= 0
\end{aligned} \tag{3.2}$$

where $\delta = (\delta_x, \delta_y)$ are the nodal displacement vectors. The coefficients d are based on

Young's modulus E and Poisson's ratio ν as follow:

$$\begin{aligned} d_{11} &= d_{22} = \frac{E\,(1-\nu)}{(1+\nu)\,(1-2\nu)} \\ d_{12} &= d_{21} = \frac{E\nu}{(1+\nu)\,(1-2\nu)} \\ d_{33} &= \frac{E}{2\,(1+\nu)} \end{aligned} \tag{3.3}$$

We used Poisson's ratio equal to 0.25. Young's modulus was assumed to be constant throughout the computational domain and therefore cancels in Equation 3.2. The complete details on the finite element discretization of Equation 3.2 can be found in reference [20].

3.5 Algorithm Overview

The metric-based anisotropic mesh adaptation algorithm is summarized here and the mesh adaptation loop is depicted in Figure 3.2.

1. We start by generating an initial isotropic unstructured mesh using the isotropic mesh generator in GRUMMP.

2. The second step is to compute the flow solution on the initial unstructured mesh. The velocity field is obtained from solution reconstruction and integrated numerically using the adaptive fourth-order Runge-Kutta method to obtain the wake centerline.

3. The wake centerline is then inserted as an internal boundary into the initial unstructured mesh using the surface insertion algorithm as explained in Section 3.2.

4. The metric is then computed at each mesh vertex.

5. The mesh and metric are passed to the anisotropic mesh adaptation algorithm in the mesh generator to produce an anisotropic mesh. The mesh adaptation scheme uses face swapping, vertex insertion, vertex removal and smoothing operations to improve the mesh quality but preserves the wake centerline without moving it at all.

6. The adapted mesh is used to re-compute the flow solution and a new wake centerline is obtained.

7. We then morph the adapted mesh to match the new wake centerline by solving the linear elasticity equation as discussed in Section 3.4. The process from step 4 to 6 in Figure 3.2 is then repeated. If the aerodynamic coefficients (C_L, C_D) are not grid converged, we move to step 7 and repeat steps 4, 5 and 6 until the aerodynamic coefficients are converged.

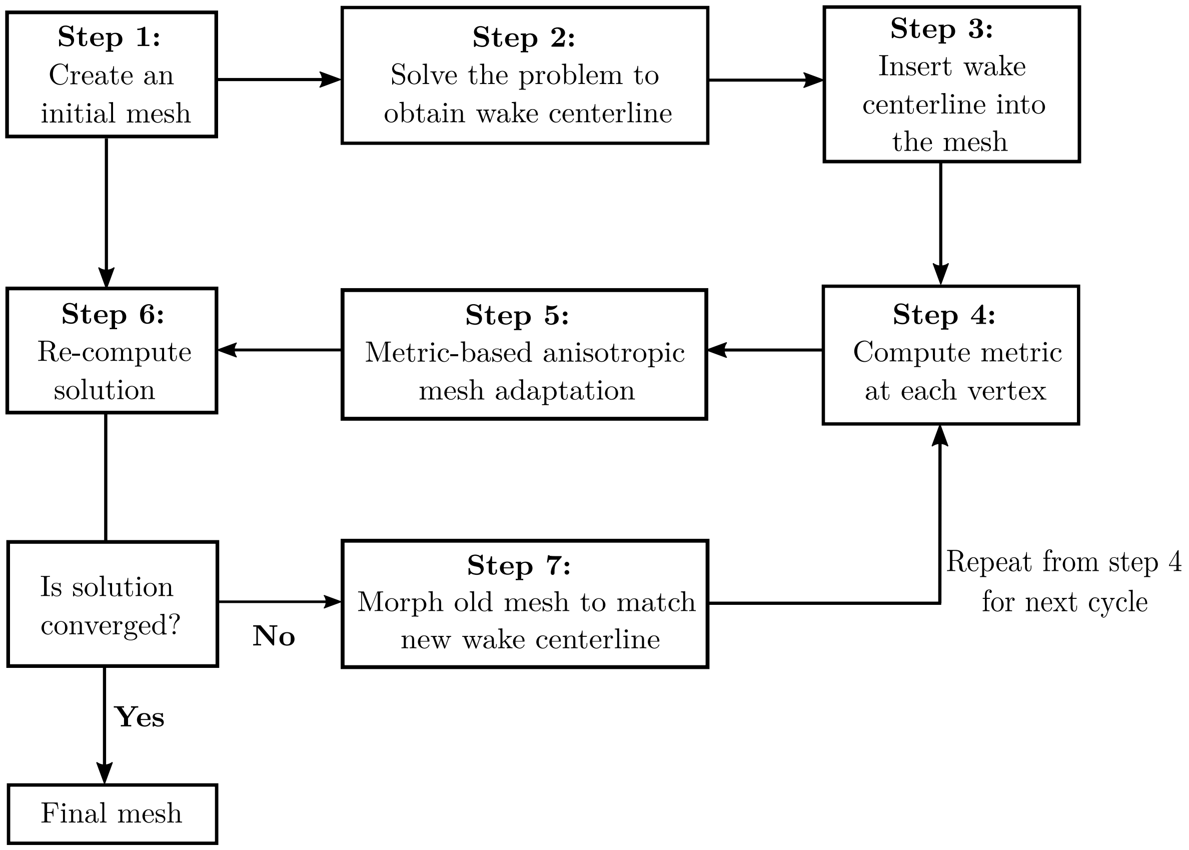

Step 1:
Create an initial mesh
Step 2:
Solve the problem to obtain wake centerline
Step 3:
Insert wake centerline into the mesh
Step 4:
Compute metric at each vertex
Step 5:
Metric-based anisotropic mesh adaptation
Step 6:
Re-compute solution
Is solution converged?
No
Step 7:
Morph old mesh to match new wake centerline
Repeat from step 4 for next cycle
Yes
Final mesh

Figure 3.2: Algorithm overview (mesh adaptation loop)

3.6 Advancing Layer Mesh Generation

Finally, another scheme is used to compare with the results obtained on the final adapted mesh using the aforementioned method. In this scheme, we use the advancing layer mesh generator of Numerow and Ollivier-Gooch [37]. We define our initial surface to include the airfoil and the discretized wake centerline as shown in Figure 3.3, and then march off the surface into the domain interior layer by layer. The wake centerline used is obtained by numerically integrating the velocity data from the initial solution on isotropic unstructured mesh. The advancing layer mesh generator produces mixed element meshes that are quad dominant and have a few triangular cells. This leads to several advantages such as: (1) lower cell count, (2) better alignment and orthogonality, (3) more geometrically similar cells, and (4) excellent resolution of the boundary layer and wake.

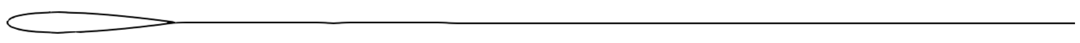


Figure 3.3: Initial geometry to generate advancing layer mesh

The advancing layer mesh is generated taking into account the flow physics. We consider the Blasius boundary layer solution for a flat plate and the two-dimensional asymptotic solution in the wake.

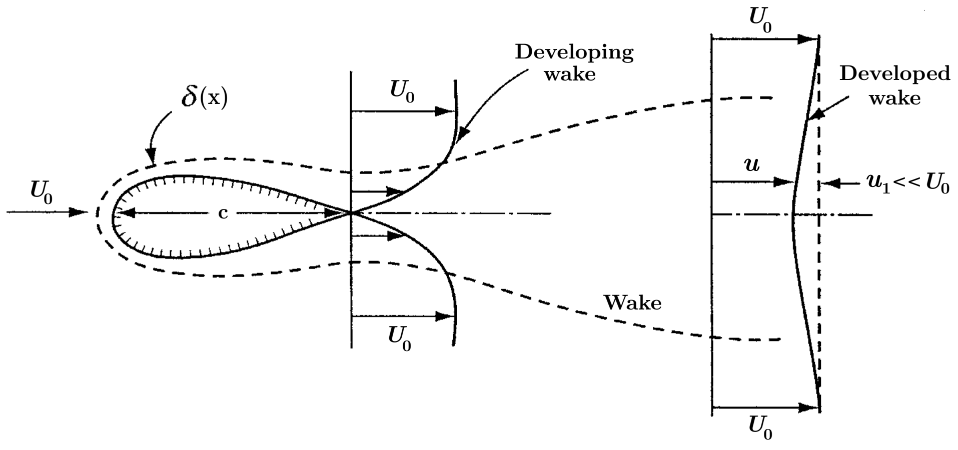


Figure 3.4: Development of wake behind an airfoil [59]

For laminar flow over a flat plate, the Blasius boundary layer solution is given by:

$$\delta_{BL}(x) \approx \frac{5x}{\sqrt{Re_x}} \tag{3.4}$$

The self-similar solution for a laminar wake is given in monograph by Berger [5] and is of the form:

$$u_1 = \frac{A}{\sqrt{\pi}\delta_w} exp\left(-\frac{y^2}{\delta_w^2}\right) \tag{3.5}$$

where A is a constant and $\delta_w(x) = 2\left(\frac{\nu x}{U_0}\right)$

Then, δ_w in the laminar wake can be related to Reynolds number as:

$$\delta_w(x) = \frac{2x}{\sqrt{Re_x}} \tag{3.6}$$

Nevertheless, using Equations 3.4 and 3.6, it is not possible to match the boundary layer thickness $\delta_{BL}(x)$ and the wake thickness $\delta_w(x)$ at the trailing edge of the airfoil. This is because $\delta_{BL}(x)$ from the laminar boundary layer solution will be greater than $\delta_w(x)$ from the laminar wake solution at the trailing edge. Figure 3.4 suggests $\delta(x)$ should keep increasing as we move downstream from the trailing edge. This means that Equation 3.6 is not valid at the trailing edge and in the near wake region, and in fact Tollmien [53] claims that the asymptotic solution given by Equation 3.5 is good only for far-wake ($x > 3c$). As a result, Equation 3.6 is also good only for the far-wake. Therefore, to generate the quad dominant advancing layer mesh, we take the average value of Equations 3.4 and 3.6 and consider a Reynolds number based on x, giving us Equation 3.7. Using Equation 3.7 ensures that $\delta(x)$ increases as we move from the leading edge of the airfoil to the trailing edge and then in the wake region downstream.

$$\delta\left(x\right)=\frac{3.5x}{\sqrt{Re_x}} \tag{3.7}$$

Equation 3.7 gives an approximate thickness $\delta\left(x\right)$ for the shear layer both on the airfoil and in the wake. In order to properly resolve the boundary layer and wake, many layers (between 20 and 30 layers) of mesh cells are required across the thickness $\delta\left(x\right)$ and so the off-wall spacing in the mesh should be much smaller than $\delta\left(x\right)$. Figure 3.5 shows the number of layers across the thickness $\delta\left(x\right)$ with variable spacing. Increasing the number of layers increases the resolution within that region and so we can derive a formula to relate the thickness $\delta\left(x\right)$, number of layers and the stretching factor for variable spacing between layers. The thickness δ is given by:

$$\begin{aligned}
\delta &= \sum_{n=1}^{N} h_n \\
&= dy_{min}\left(1+s+s^2+...+s^{N-1}\right) \\
&= dy_{min}\sum_{n=1}^{N} s^{n-1} \\
&= dy_{min}\frac{\left(1+s+s^2+...+s^{N-1}\right)(1-s)}{(1-s)} \\
\delta &= dy_{min}\frac{(s^N-1)}{(s-1)} \qquad (3.8)
\end{aligned}$$

Then from Equations 3.7 and 3.8, we have

$$dy_{min} = \frac{3.5x}{\sqrt{Re_x}}\frac{(s-1)}{(s^N-1)} \tag{3.9}$$

where dy_{min} is the off-wall spacing, s is the stretching factor $(s>1)$ and N is the number of layers across the thickness δ. As per Equation 3.9, off-wall spacing is based on x and the Reynolds number Re_x.

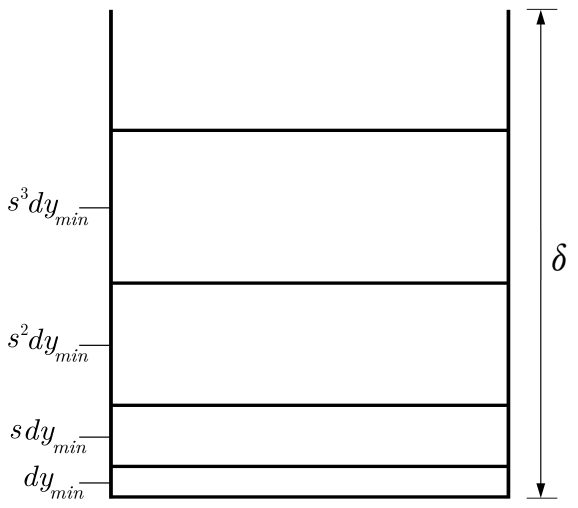


Figure 3.5: Number of layers across thickness $\delta\,(x)$

Chapter 4

Results

In this chapter, we first present results to verify the correctness of the streamline tracking and mesh morphing algorithms. Later, different laminar flow cases are included to demonstrate our approach for mesh adaptation. For each of these test cases, the actual velocity data from the flow solver was used and integrated numerically. The solution was computed based on the second-order accurate vertex-centered finite volume solver as discussed by Ollivier-Gooch in his paper [40]. This flow solver makes use of least squares reconstruction [39], Roe's scheme [46] and the Newton-GMRES method [34, 36] for fast steady-state convergence. Moreover, the results of test cases from adapted triangular meshes with and without the wake centerline and advancing layer meshes are compared.

The laminar flow results include both attached and separated flow cases. The flow conditions for the first two cases are: (1) $Re = 2100$, $Ma = 0.5$, $\alpha = 0^\circ$ and (2) $Re = 2100$, $Ma = 0.5$, $\alpha = 2^\circ$ and for these cases, we choose $Re = 2100$ as the flow is attached at these conditions. Moreover, the third test is chosen to show that our algorithm works for separated flows as well and also there are numerical results available by well-established CFD codes for $Re = 5000$, $Ma = 0.5$ and $\alpha = 0^\circ$.

In the end, we also present a test case for turbulent subsonic flow. For turbulent flow, the solution was computed using the second-order accurate cell-centered finite volume solver and the velocity field was integrated to track just the wake centerline at different angles of attack, without doing any mesh adaptation.

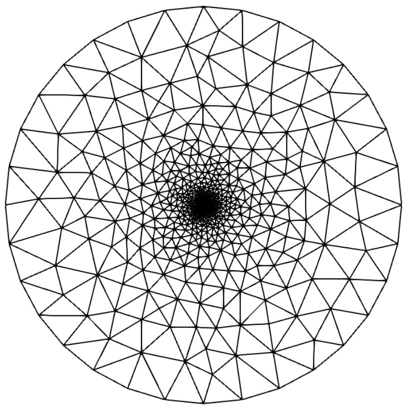
4.1 Verification Cases

4.1.1 Streamline Tracking Algorithm

The streamline tracking algorithm was tested using an analytical velocity field given by:

$$\frac{d}{dt}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -2y \\ x \end{pmatrix} \tag{4.1}$$

Integrating the above velocity field analytically gives streamlines that are ellipses. The problem was solved numerically on an isotropic unstructured mesh around the NACA 0012 airfoil. Figure 4.1 shows the mesh, which was generated using the isotropic mesh generator in GRUMMP [38]. The far-field boundary was a circle with radius $500c$ centered at the leading edge of the airfoil. The trailing edge of the airfoil, $(1, 0)$, was chosen as the starting point to track the streamline. The velocity data was control volume averaged onto the mesh, then reconstructed. Because the velocity is linear, this reconstruction was exact. Nevertheless, this test verifies our ability to track streamlines through a mesh using reconstructed velocity data.


(a) Far-field view

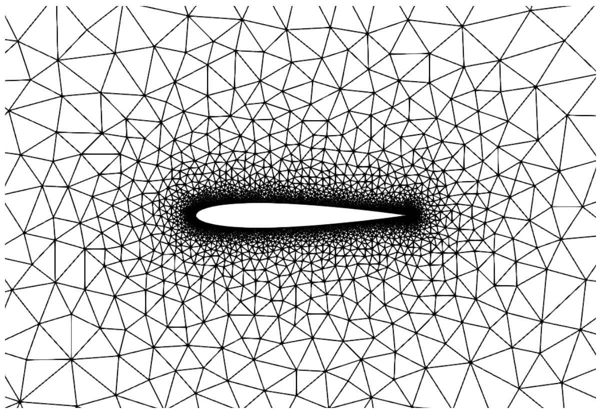

(b) Closeup view

Figure 4.1: Isotropic unstructured mesh around a NACA 0012 airfoil with 7310 cells and 3871 vertices

The streamline was integrated halfway around the ellipse. Finally, to validate the result, the case was compared to the Paraview streamtracer result as shown in Figure 4.2.

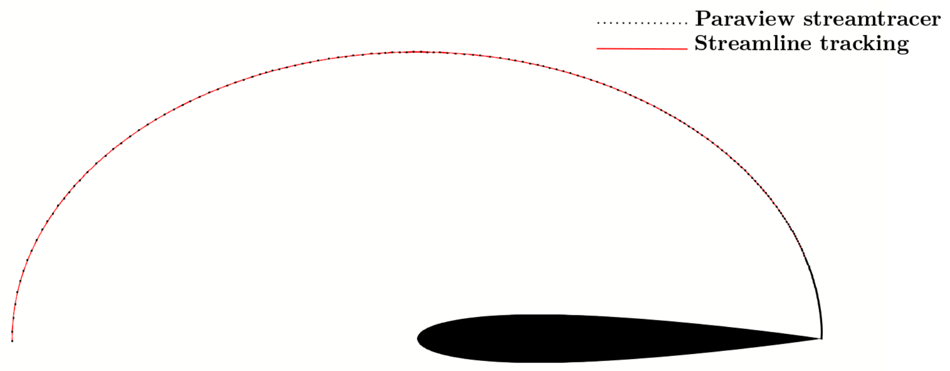


Figure 4.2: Streamline tracking for an analytical velocity field

Next, the order of accuracy of the numerical method was tested based on the streamline data obtained from numerical integration. The velocity field was integrated numerically using three different time-steps to obtain three sets of data for streamline location. Then, the error between the actual and numerically predicted location on the streamline at an instant in time was calculated as:

$$\text{error} = \sqrt{(x - x_n)^2 + (y - y_n)^2}$$

where x and y are the analytical values while x_n and y_n are numerical values.

Time-Step	error	ratio $\left(\frac{(error)_{\triangle t=h}}{(error)_{\triangle t=h/2}}\right)$
0.05	4.601×10^{-7}	–
0.025	2.870×10^{-8}	16.031
0.0125	1.800×10^{-9}	15.944

Table 4.1: Error in solution (values at $t = 2.20$)

Table 4.1 illustrates that the error drops by 16 times when the time-step is reduced by half so the Runge-Kutta method is fourth-order accurate as expected. Moreover, Figure 4.3 shows the error in solution at each point along the curve for three different time-steps mentioned above. The ratio of error between the green and red curves and between the red and blue curves at each point was found to be almost exactly 16.

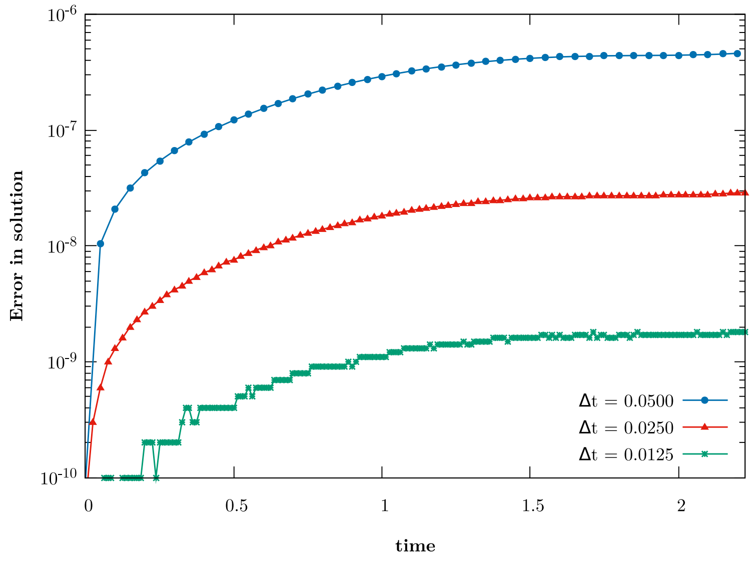


Figure 4.3: Error in solution at each point along the curve

4.1.2 Mesh Morphing Algorithm

To verify the correctness of the mesh morphing approach, we used the method of manufactured solutions. The nodal displacement vectors were selected as:

$$\begin{aligned}\delta_x &= \sin\left(\pi x + \frac{\pi}{2}\right)\sin\left(\pi y + \frac{\pi}{2}\right) \\ \delta_y &= \sin\left(\pi x + \frac{\pi}{2}\right)\sinh\left(\pi y + \frac{\pi}{2}\right)\end{aligned} \tag{4.2}$$

The manufactured solution is substituted into the linear elasticity equations (Equation 3.2) and a source term vector was identified (Equation 4.3).

$$S = \begin{pmatrix} S_x \\ S_y \end{pmatrix} \tag{4.3}$$

where

$$\begin{aligned}S_x &= d_{11}\sin\left(\pi x + \frac{\pi}{2}\right)\pi^2\sin\left(\pi y + \frac{\pi}{2}\right) - d_{12}\cos\left(\pi x + \frac{\pi}{2}\right)\pi^2\cosh\left(\pi y + \frac{\pi}{2}\right) \\ &\quad -d_{33}\left(-\sin\left(\pi x + \frac{\pi}{2}\right)\pi^2\sin\left(\pi y + \frac{\pi}{2}\right) + \cos\left(\pi x + \frac{\pi}{2}\right)\pi^2\cosh\left(\pi y + \frac{\pi}{2}\right)\right)\end{aligned}$$

and

$$\begin{aligned} S_y &= -d_{33}\left(\cos\left(\pi x+\frac{\pi}{2}\right)\pi^2\cos\left(\pi y+\frac{\pi}{2}\right)-\sin\left(\pi x+\frac{\pi}{2}\right)\pi^2\sinh\left(\pi y+\frac{\pi}{2}\right)\right)\\ &\quad -d_{21}\cos\left(\pi x+\frac{\pi}{2}\right)\pi^2\cos\left(\pi y+\frac{\pi}{2}\right)-d_{22}\sin\left(\pi x+\frac{\pi}{2}\right)\pi^2\sinh\left(\pi y+\frac{\pi}{2}\right)\end{aligned}$$

The discrete problem was solved with this source term. The source term vector was integrated using the Gauss quadrature rule. The L_2 norm of error was then computed as the difference between the manufactured solution of Equation 4.2 and the linear finite element numerical solution. Figure 4.4 shows the variation of L_2 norm of error in solution with the mesh size and confirms that the numerical solution is second-order accurate for linear elements, as expected.

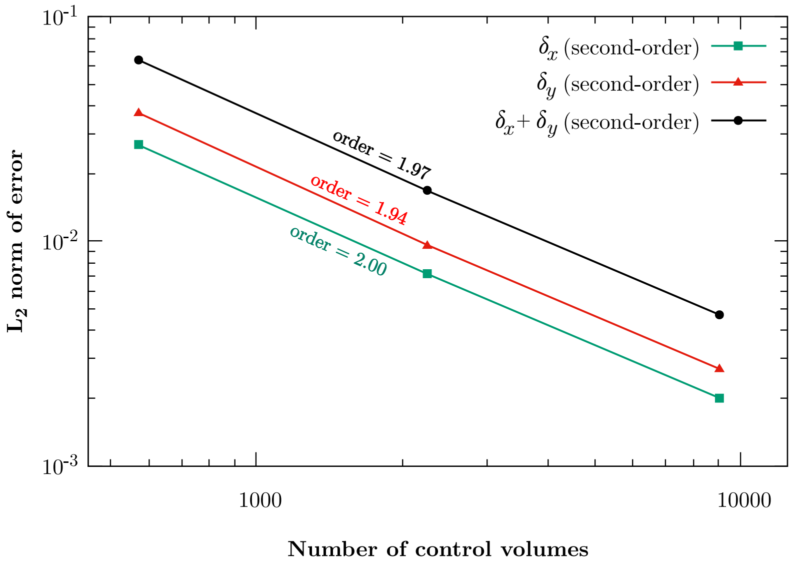
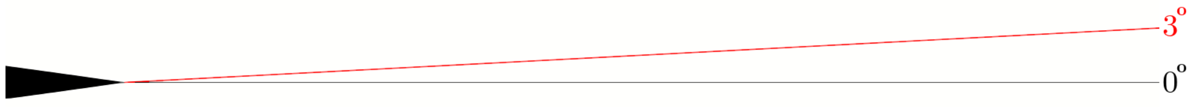
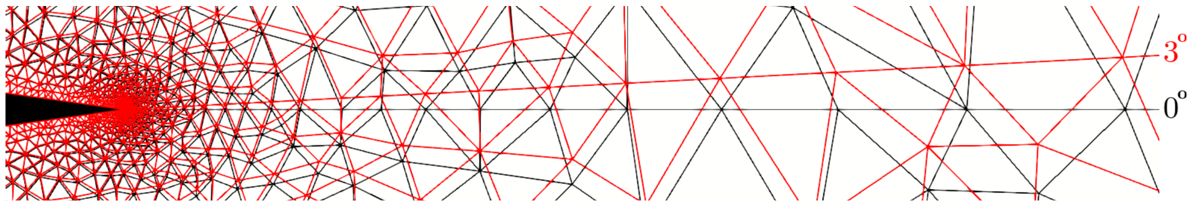


Figure 4.4: L_2 norm of error for linear elasticity

The correct implementation of boundary condition to displace the wake centerline was also tested by inserting an imaginary horizontal line at the trailing edge of the airfoil in an unstructured mesh and then displacing it from 0° to 3° as shown in Figure 4.5. This test confirms that the mesh morphing algorithm displaces the inserted line as required and morphs the mesh accordingly.



(a) Line at 0° (black) and 3° (red)



(b) Mesh before (black) and after (red) mesh morphing

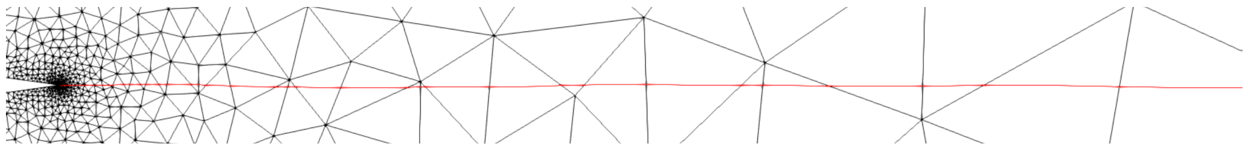
Figure 4.5: Mesh morphing to displace horizontal line from 0° to 3°

4.2 Laminar Flow Test Cases

4.2.1 Viscous flow at $Re = 2100$, $Ma = 0.5$ and $\alpha = 0°$

Our first adaptation test is subsonic viscous flow at Reynolds number $Re = 2100$, Mach number $Ma = 0.5$ and angle of attack $\alpha = 0°$. The viscous terms are discretized as discussed by Ollivier-Gooch and Van Altena [39]. Using the mesh in Figure 4.1, we computed the initial flow solution and integrated the velocity field to track the streamline from the trailing edge of NACA 0012 airfoil. The streamline was then inserted as an internal boundary into the same unstructured mesh using the surface insertion algorithm. It can be seen from Figures 4.6a[2] and 4.6b below that the result of Paraview streamtracer matches with the streamline tracking algorithm result. The wake centerline is not exactly straight but the oscillations are of the order 10^{-3} to 10^{-5} chords from the symmetry axis; these oscillations become smaller on more refined meshes and as we go further downstream from the trailing edge.

[2]In Figure 4.6a, red line is the position of streamline obtained from Paraview streamtracer and so it is not a part of the mesh.


(a) Paraview streamtracer result (Mesh before streamline insertion with 7310 cells) - Viscous flow - Red line is the streamline

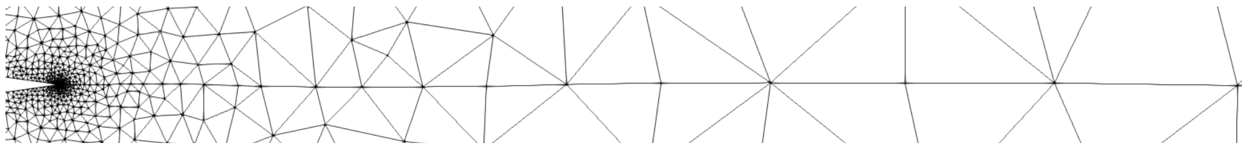

(b) Streamline tracking algorithm result (Mesh after streamline insertion with 7468 cells) - Viscous flow

Figure 4.6: Streamline tracking at $Re = 2100$, $Ma = 0.5$ and $\alpha = 0^{\circ}$

A metric was computed for the mesh in Figure 4.6b and passed to the anisotropic mesh generation algorithm in GRUMMP to do mesh adaptation. The adapted mesh was used to compute the flow solution again and obtain the new wake centerline. We then morphed the adapted mesh to match the new wake centerline by solving the linear elasticity problem as discussed in mesh morphing section in Chapter 3. Figure 4.7 shows the mesh before and after mesh morphing. This process was repeated for subsequent cycles of anisotropic mesh adaptation and helped to improve the solution accuracy. The flow solution in Figure 4.8a on the initial mesh shows that the mesh is not well-suited to compute the viscous flow and resolve the boundary layer and wake. However, after each adaptation, better resolution and mesh alignment is obtained in the boundary layer and wake as shown in Figure 4.8.

The position of the wake centerline on the initial and final adapted mesh is depicted in Figure 4.9. The wake centerline from the final adapted mesh is closer to the symmetry axis, illustrating that the final mesh can track the position of wake better for this symmetric flow. The small variations are still present since the mesh is not completely symmetric.

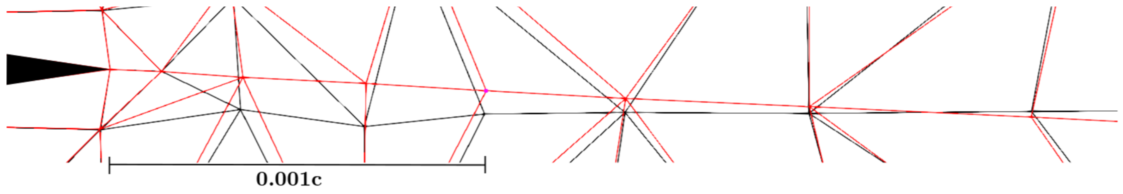
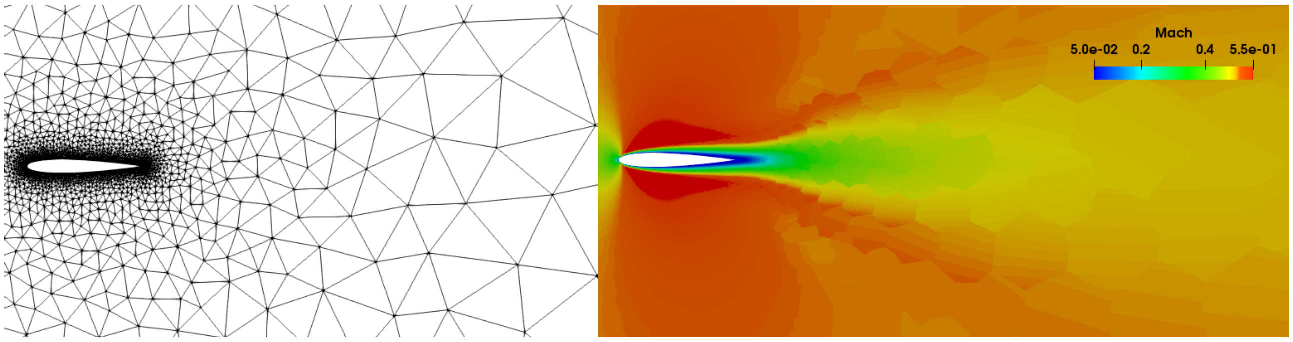
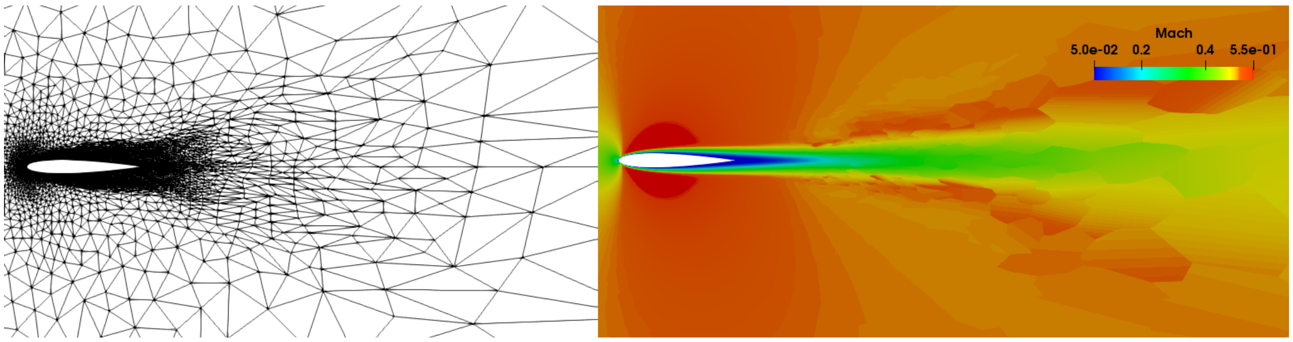


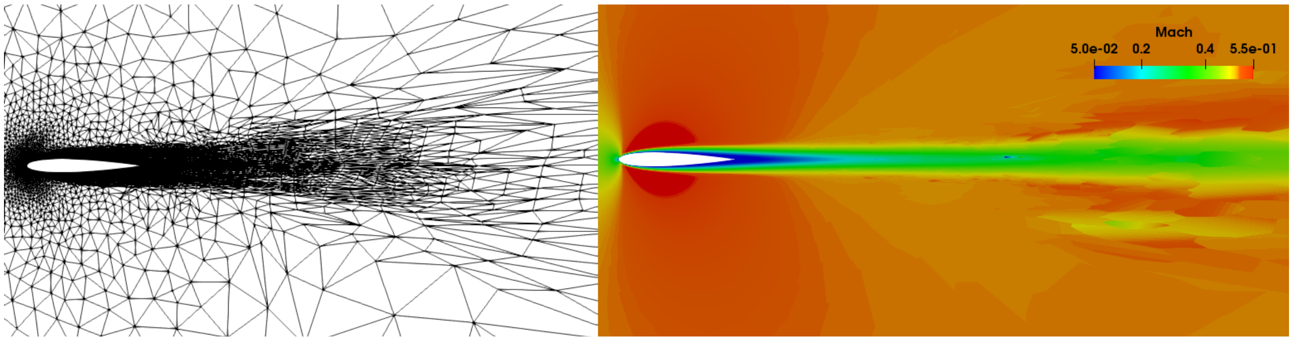
Figure 4.7: First adapted mesh (black) and morphed mesh (red) before second adaptation



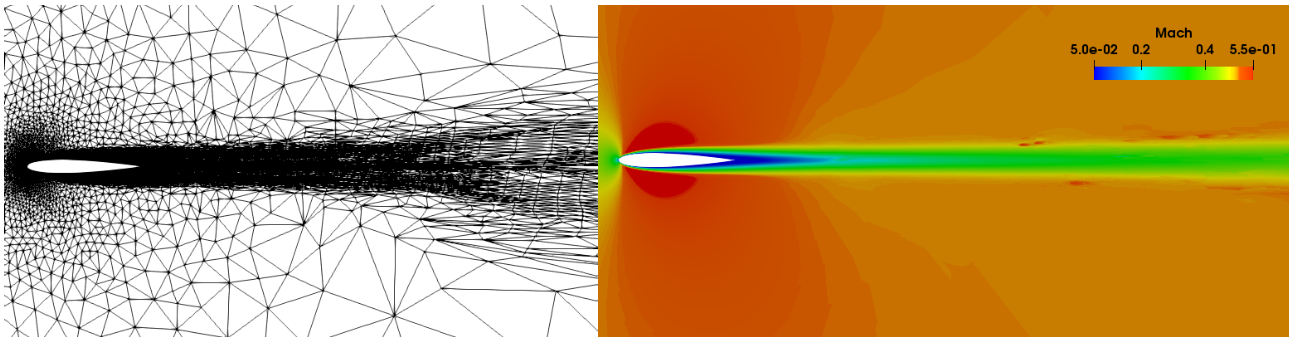
(a) Solution on initial mesh (3871 vertices)



(b) Solution after first adaptation (6048 vertices)



(c) Solution after second adaptation (8268 vertices)



(d) Solution on final mesh (12121 vertices)

Figure 4.8: Adapted meshes and flow solutions with wake centerline at $Re = 2100$, $Ma = 0.5$ and $\alpha = 0^{\circ}$

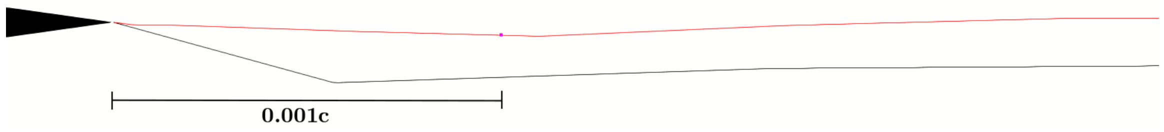


Figure 4.9: Wake centerline position - initial (black) and final (red) at $Re = 2100$, $Ma = 0.5$ and $\alpha = 0^{\circ}$ - Closeup view at the trailing edge

After obtaining the solutions on triangular meshes, an anisotropic quad dominant mesh was generated using the advancing the layer mesh generator in GRUMMP. The initial geometry used to generate the advancing layer mesh included the NACA 0012 airfoil and the discretized wake centerline obtained from an initial solution in Figure 4.8a.

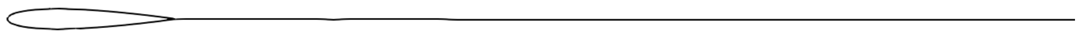
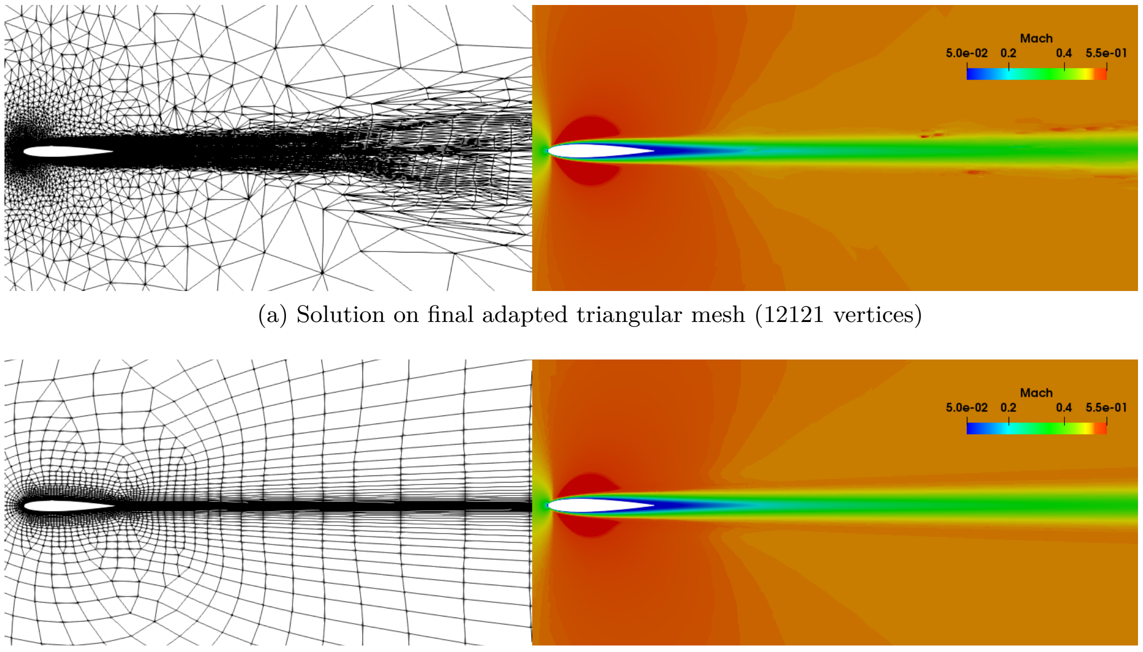


Figure 4.10: Initial geometry for advancing layer mesh at $Re = 2100$, $Ma = 0.5$ and $\alpha = 0^{\circ}$



(a) Solution on final adapted triangular mesh (12121 vertices)

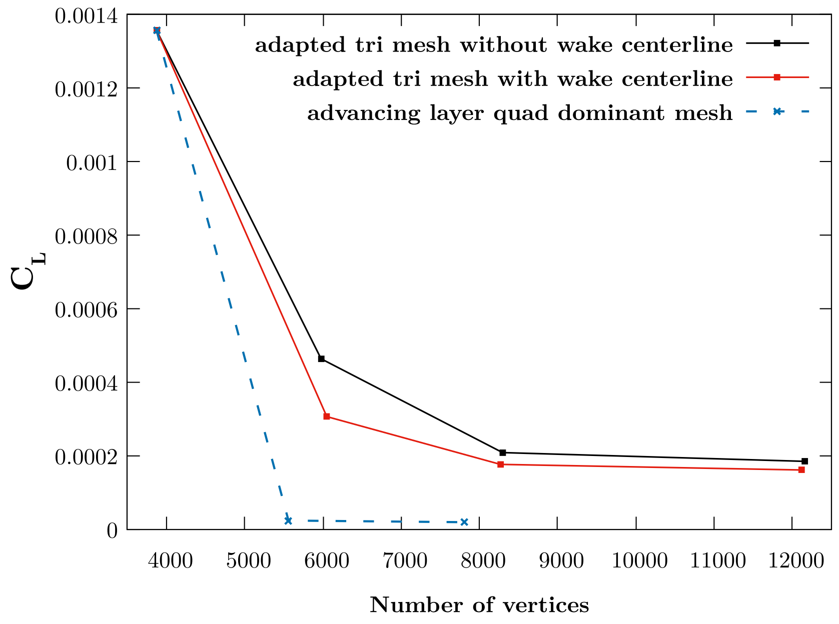
(b) Solution on advancing layer quad dominant mesh (5558 vertices)

Figure 4.11: Flow solutions on final adapted triangular mesh and advancing layer mesh at $Re = 2100$, $Ma = 0.5$ and $\alpha = 0^{\circ}$, showing better solution behavior in the wake for the advancing layer mesh

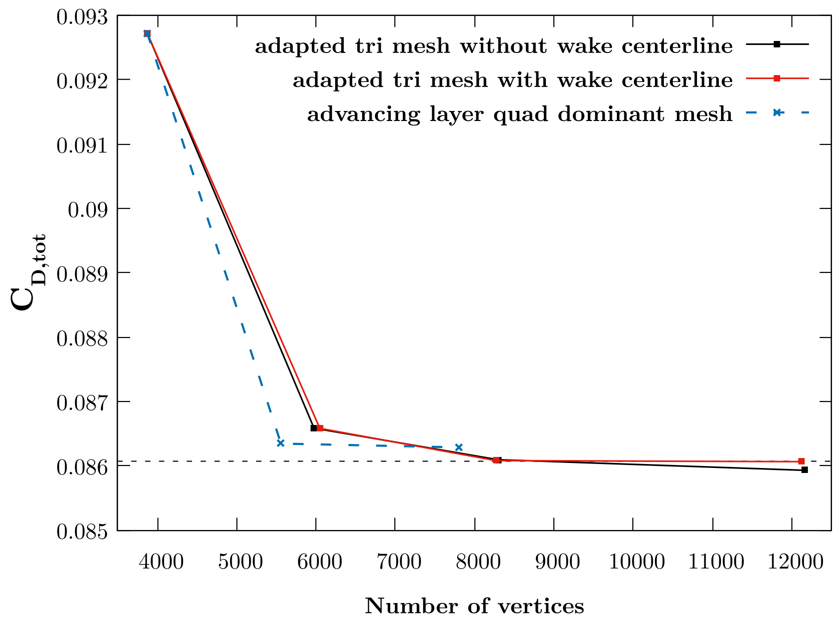
Figure 4.11 shows the solution on the final adapted triangular mesh and the quad dominant advancing layer mesh, illustrating that the advancing layer mesh with many fewer degrees of freedom can track the wake for a longer distance compared to adapted triangular meshes.

The convergence history of aerodynamic coefficients with respect to degrees of freedom for triangular meshes with and without the wake centerline and for the advancing layer quad dominant meshes is displayed in Figure 4.12. For triangular meshes, it can be observed that the lift coefficient for meshes with the wake centerline included converges slightly faster, while the convergence of drag coefficients is effectively identical. Moreover, the solution is converging to a single value on meshes with and without the wake centerline. By comparison, for the advancing layer meshes, the aerodynamic coefficients converge faster compared to both triangular meshes. Especially, the lift coefficient on the advancing layer meshes converges much faster and has a smaller error as we expect to get $C_L = 0$ for this symmetric flow. The difference in drag coefficient from the refined advancing layer quad dominant mesh and the grid converged value using triangular meshes is just 0.26%.

In Figure 4.13, we present the convergence history against total CPU time for meshing and solver. The total CPU time for triangular meshes with a wake centerline is somewhat less compared to meshes without the wake centerline. There is clearly a CPU time advantage for the advancing layer quad dominant meshes compared to both triangular meshes. The main reason is that it is much cheaper to generate advancing layer mesh compared to doing mesh adaptation and this reduces CPU time required for meshing. Also, the advancing layer mesh can track the wake much better with fewer degrees of freedom which reduces the CPU time needed by the solver.

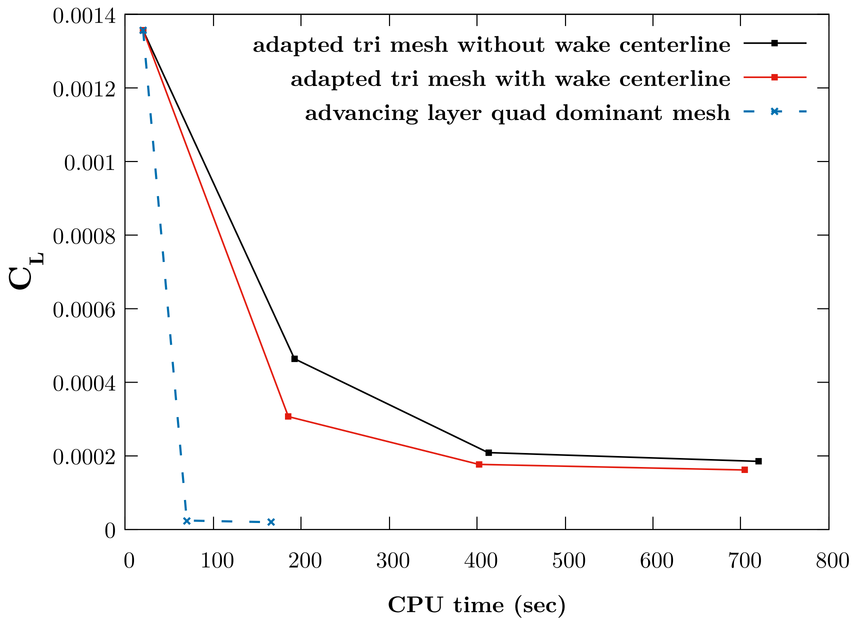


(a) Lift coefficient

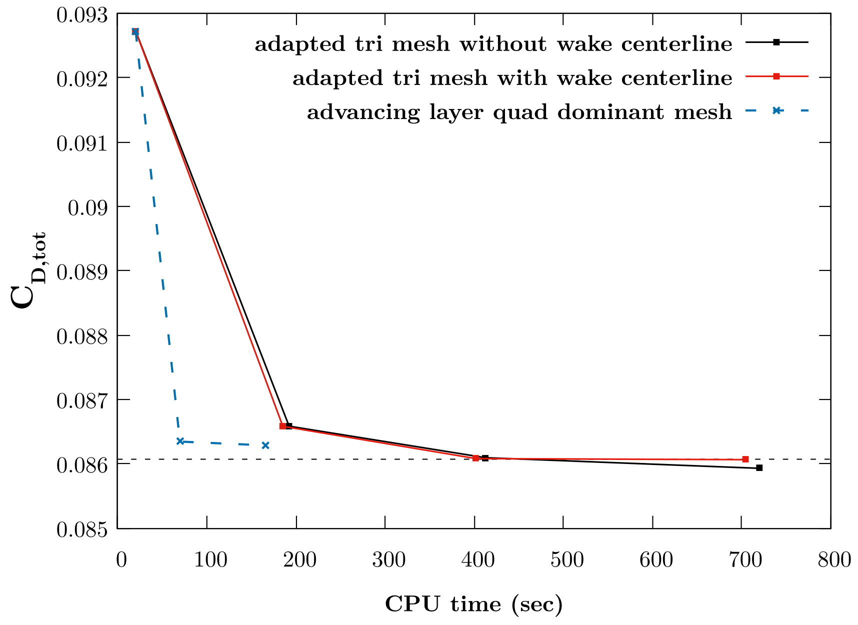


(b) Drag coefficient

Figure 4.12: Convergence of lift and drag coefficients against degrees of freedom (vertices) at $Re = 2100$, $Ma = 0.5$ and $\alpha = 0^{\circ}$

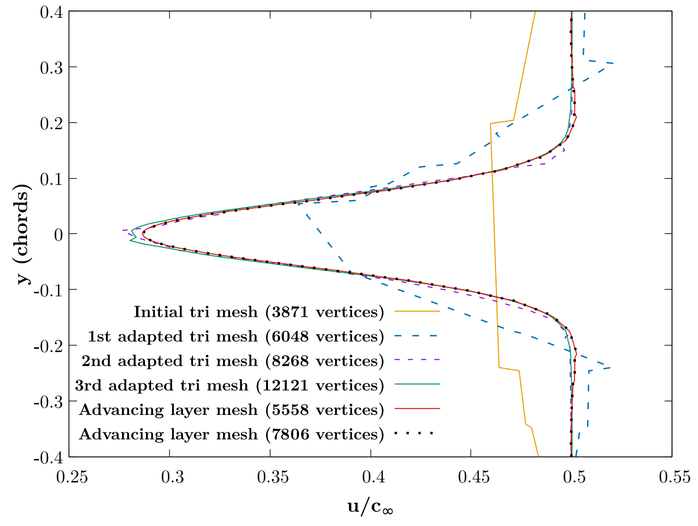


(a) Lift coefficient

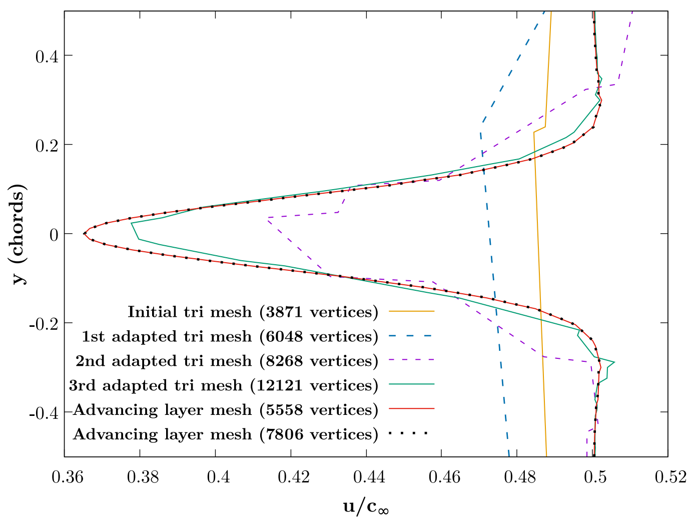


(b) Drag coefficient

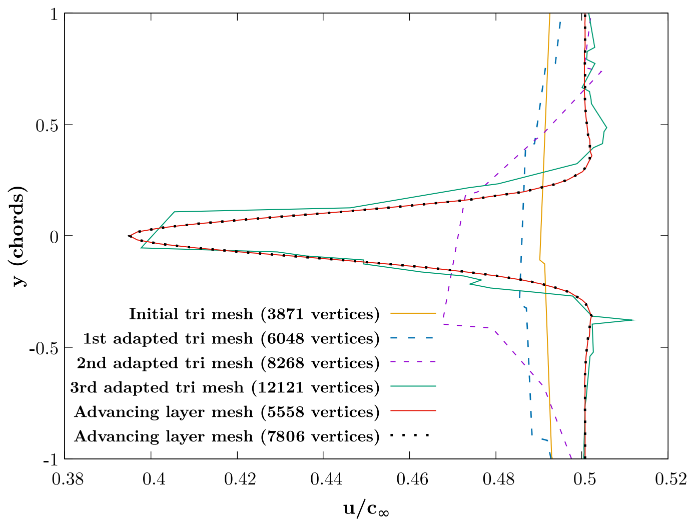
Figure 4.13: Convergence of lift and drag coefficients against total CPU time for meshing and solver at $Re = 2100$, $Ma = 0.5$ and $\alpha = 0^{\circ}$



(a) Two chords downstream of the trailing edge



(b) Five chords downstream of the trailing edge



(c) Eight chords downstream of the trailing edge

Figure 4.14: Velocity profiles in the wake for triangular and advancing layer quad dominant meshes

In Figure 4.14, we present the comparison of velocity profiles in the wake for triangular meshes with wake centerline and the advancing layer quad dominant meshes at different locations downstream from the trailing edge. Figures 4.14a to 4.14c illustrate that there is a velocity deficit in the wake as expected. Furthermore, we expect to get symmetric velocity profiles in the wake for this case. At each location (2, 5 and 8 chords downstream the trailing edge), the velocity profiles from the advancing layer meshes are almost symmetric. The velocity profile from the initial triangular mesh is not physical due to poor mesh resolution but as we do mesh adaptation, the velocity profile improves with each mesh refinement and approaches the velocity profiles from the advancing layer meshes. At two chords downstream of the trailing edge, the final adapted mesh solution is nearly as smooth and symmetric as the solution on the advancing layer meshes. Note that compared to the advancing layer meshes, the final adapted triangular mesh has much more the degrees of freedom. Also, the velocity profile from the adapted meshes gets worse in comparison as we move further downstream from the trailing edge from two chords to eight chords. Therefore, this comparison shows that the advancing layer meshes can resolve the wake more efficiently. The velocity profiles from the advancing layer quad dominant meshes with 5558 and 7806 vertices are identical, illustrating that increasing the mesh resolution does not change the velocity profiles further.

4.2.2 Viscous flow at $Re = 2100$, $Ma = 0.5$ and $\alpha = 2^\circ$

The second test case is viscous flow around the NACA 0012 airfoil at a non-zero angle of attack. All other flow conditions and initial mesh used were similar to first test case. Figure 4.15 shows the streamline tracking algorithm result for this case. The steps used for anisotropic mesh adaptation were same but here four cycles of adaptation were required for grid convergence of aerodynamic coefficients. Figure 4.16 shows the mesh morphing results.

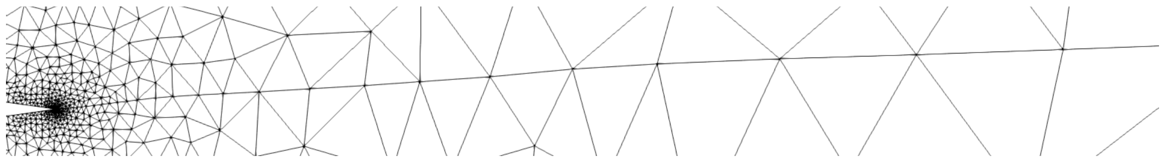


Figure 4.15: Streamline tracking algorithm result at $Re = 2100$, $Ma = 0.5$ and $\alpha = 2^\circ$

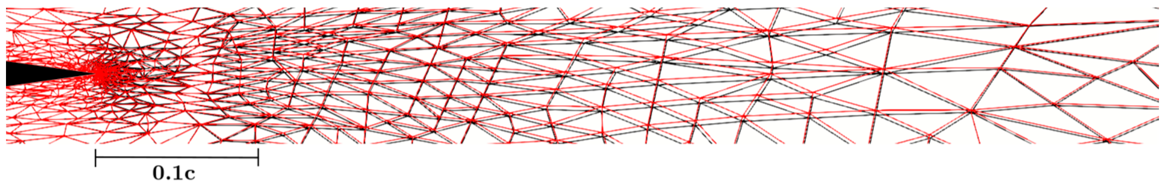
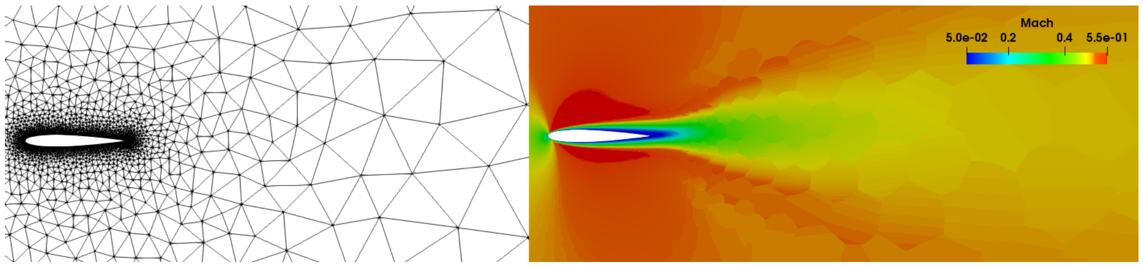
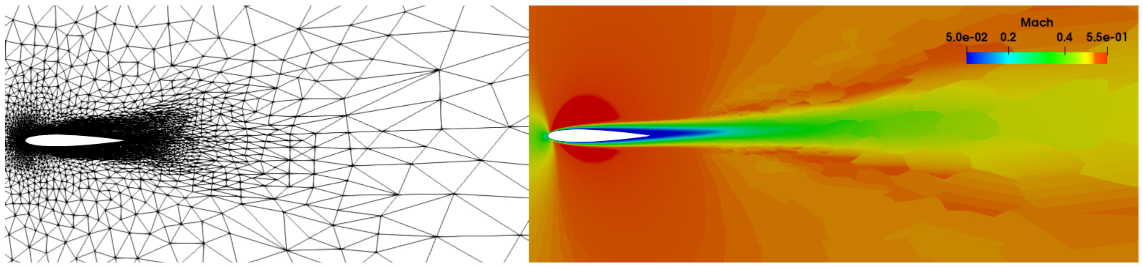


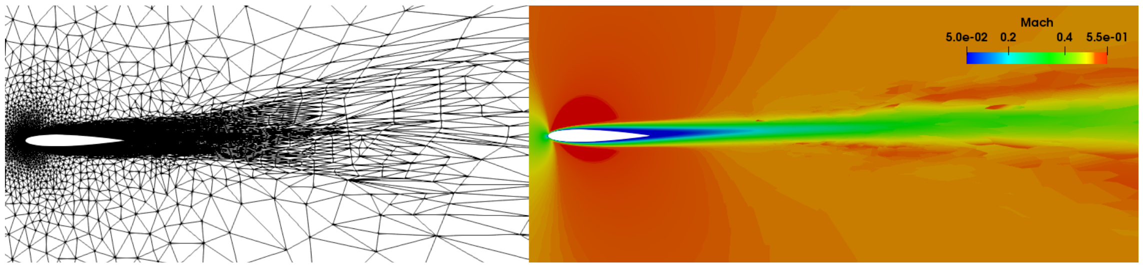
Figure 4.16: First adapted mesh (black) and morphed mesh (red) before second adaptation



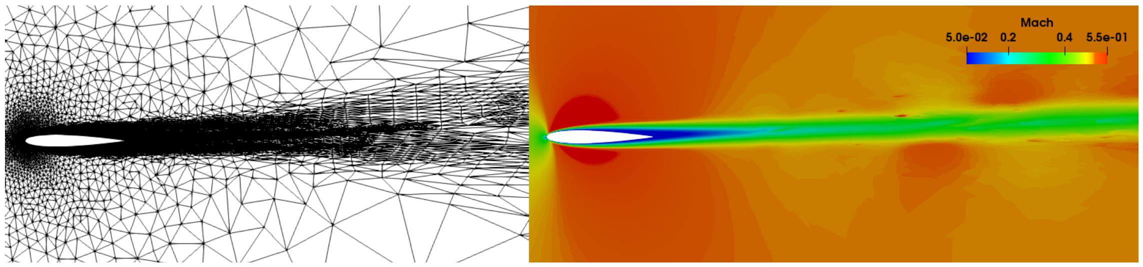
(a) Solution on initial mesh (3871 vertices)



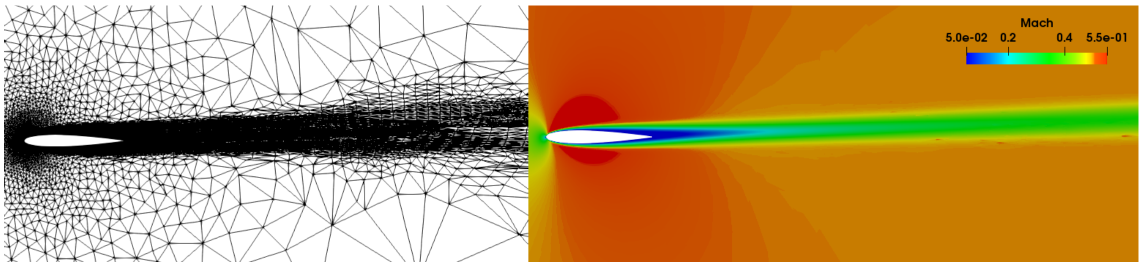
(b) Solution after first adaptation (6038 vertices)



(c) Solution after second adaptation (8584 vertices)



(d) Solution after third adaptation (10667 vertices)



(e) Solution on final mesh (13849 vertices)

Figure 4.17: Adapted meshes and flow solutions with wake centerline at $Re = 2100$, $Ma = 0.5$ and $\alpha = 2^{\circ}$

The meshes and flow solutions are depicted in Figure 4.17 and show improvement in the resolution of the boundary layer and wake after each adaptation. Figure 4.18 displays the wake centerline position on the initial and final adapted mesh.

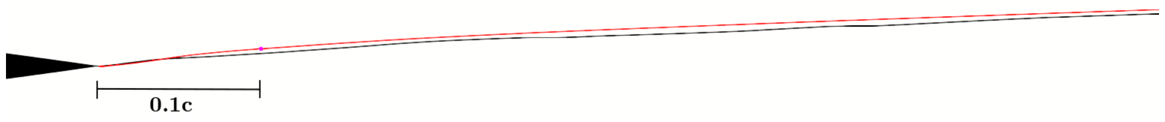


Figure 4.18: Wake centerline position - initial (black) and final (red) at $Re = 2100$, $Ma = 0.5$ and $\alpha = 2^{\circ}$ - Closeup view at the trailing edge

Next, an anisotropic quad dominant mesh was generated using the advancing layer mesh generator. The initial geometry is depicted in Figure 4.19 and consists of the NACA 0012 airfoil and the discretized wake centerline obtained from an initial solution.

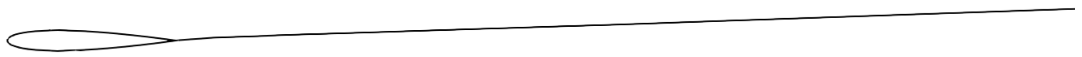
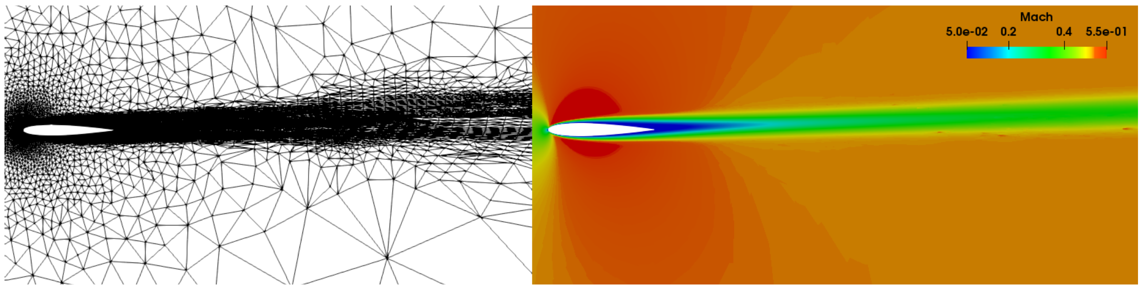
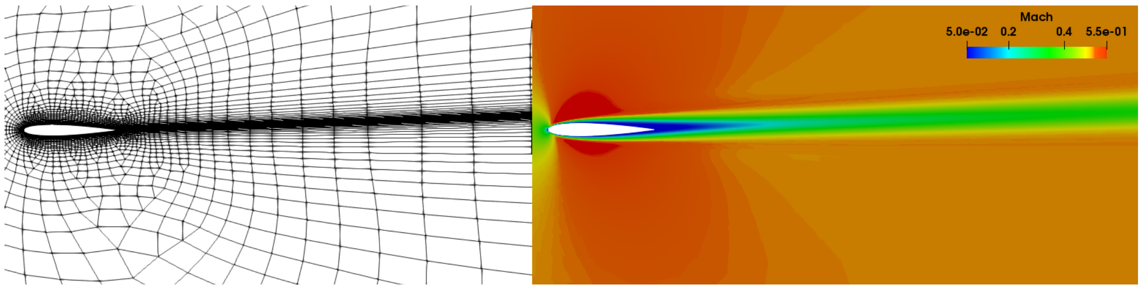


Figure 4.19: Initial geometry for quad dominant mesh at $Re = 2100$, $Ma = 0.5$ and $\alpha = 2^{\circ}$



(a) Solution on final adapted triangular mesh (13849 vertices)



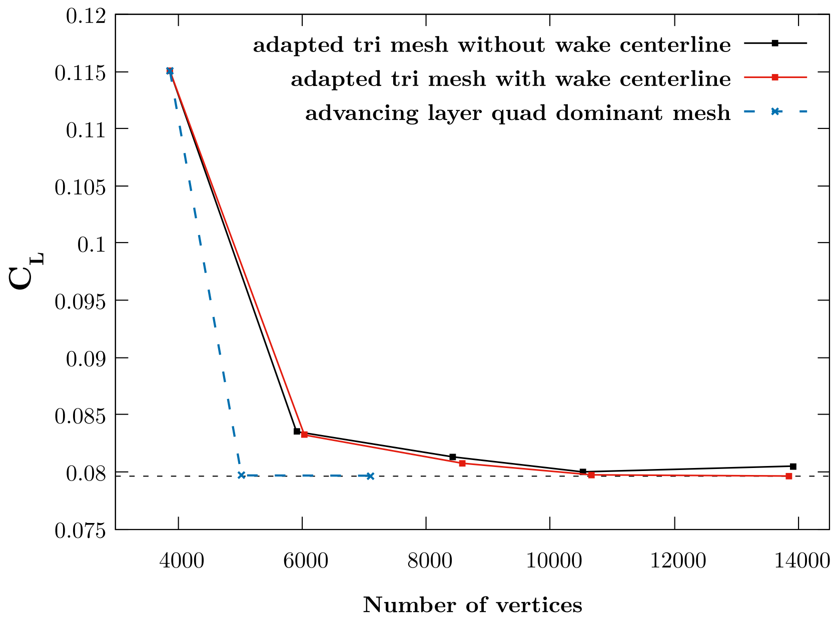
(b) Solution on advancing layer quad dominant mesh (5024 vertices)

Figure 4.20: Flow solutions on final adapted triangular mesh and advancing layer mesh at $Re = 2100$, $Ma = 0.5$ and $\alpha = 2^{\circ}$, showing better solution behavior in the wake for the advancing layer mesh

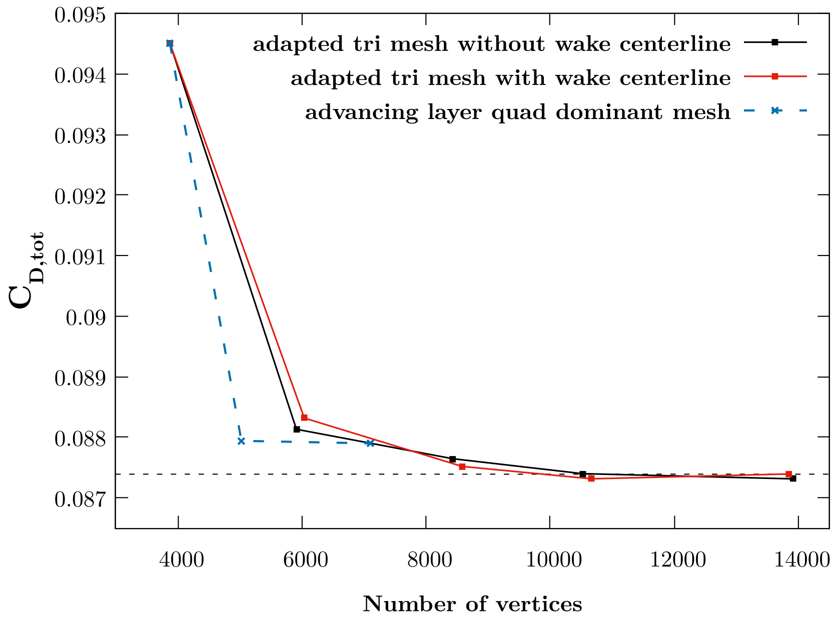
The flow solutions on the final adapted triangular and advancing layer quad dominant meshes are depicted in Figure 4.20, indicating that the advancing layer mesh can track the wake for longer distance with many fewer degrees of freedom.

The variation in aerodynamic coefficients for each mesh type is displayed in Figure 4.21. For triangular meshes, the convergence of lift and drag coefficients is effectively identical and the coefficients converge to a single value with and without the wake centerline. Using the advancing layer meshes, the aerodynamic coefficients converge faster. For example, the converged lift coefficient obtained from the advancing layer meshes is the same as the grid converged value of lift coefficient from triangular meshes, although the advancing layer meshes have many fewer degrees of freedom. For the drag coefficient, the difference between the grid converged value from triangular meshes and the advancing layer quad dominant meshes is just 0.52%.

Figure 4.22 shows the convergence of aerodynamic coefficients with respect to total CPU time for meshing and flow solution. In the case of triangular meshes, there is a CPU time advantage for the mesh with wake centerline compared to the mesh without wake centerline but just for the last cycle of mesh adaptation. However, there is clearly a CPU time advantage in using the advancing layer mesh. This is because we increased the mesh density in the desired regions and so with much fewer degrees of freedom overall, we can obtain almost the same values for aerodynamic coefficients as the grid converged values from triangular meshes.

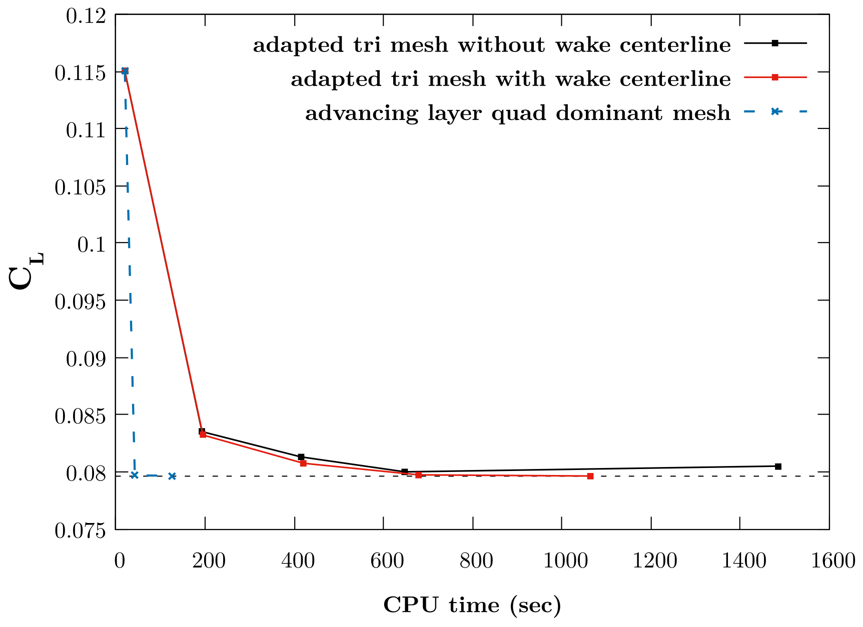


(a) Lift coefficient

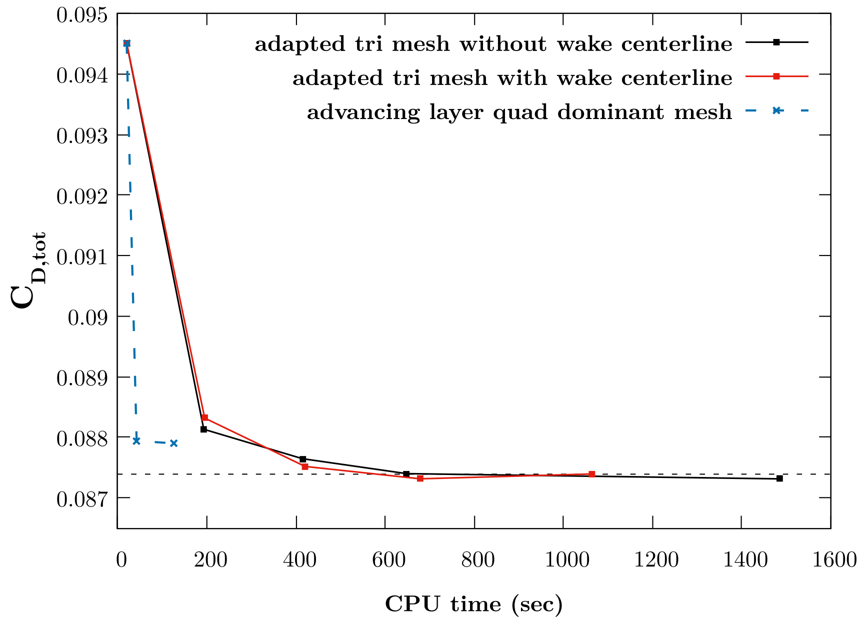


(b) Drag coefficient

Figure 4.21: Convergence of lift and drag coefficients against degrees of freedom (vertices) at $Re = 2100$, $Ma = 0.5$ and $\alpha = 2^{\circ}$

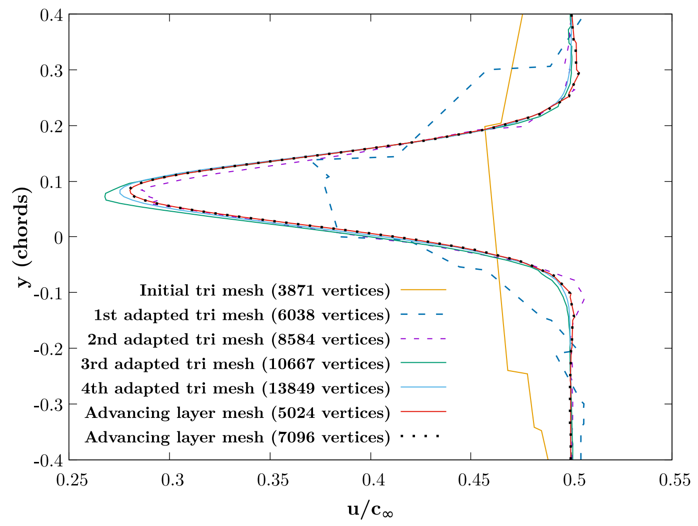


(a) Lift coefficient

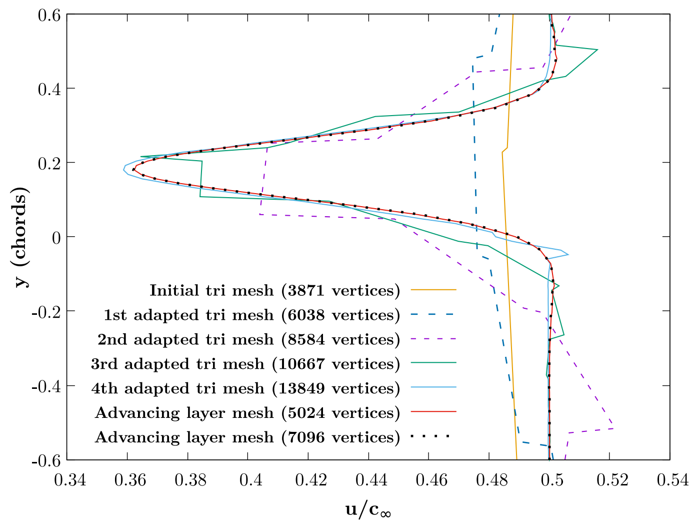


(b) Drag coefficient

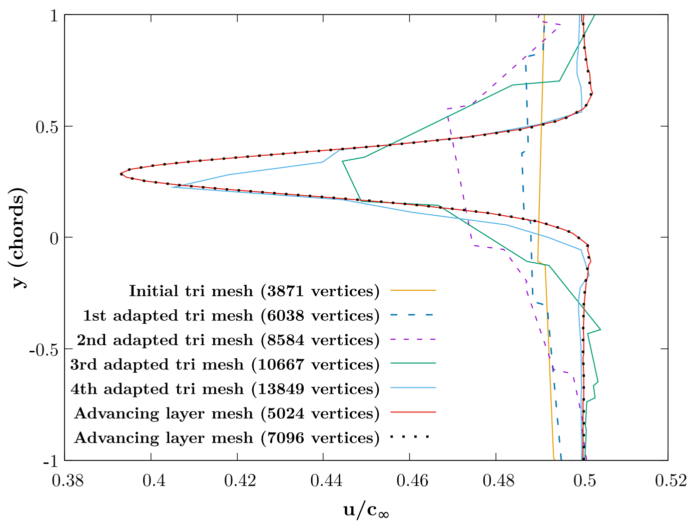
Figure 4.22: Convergence of lift and drag coefficients against total CPU time for meshing and solver at $Re = 2100$, $Ma = 0.5$ and $\alpha = 2^{\circ}$



(a) Two chords downstream of the trailing edge



(b) Five chords downstream of the trailing edge



(c) Eight chords downstream of the trailing edge

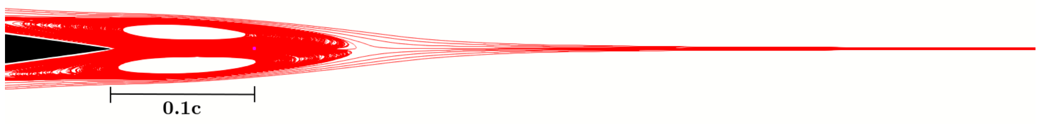
Figure 4.23: Velocity profiles in the wake for triangular and advancing layer quad dominant meshes

In Figure 4.23, we compare the velocity profiles in the wake from triangular meshes with wake centerline and the advancing layer meshes at different positions downstream from the trailing edge. As expected, there is a velocity deficit in the wake. With many fewer degrees of freedom, the velocity profiles from the advancing layer meshes are better for all three positions downstream the trailing edge. Also, the velocity profiles from advancing layer meshes with 5024 vertices and 7096 vertices are identical. However, with each mesh adaptation, the velocity profiles from the triangular meshes get closer to matching the advancing layer quad dominant meshes. The advancing layer meshes have much fewer degrees of freedom compared to the fourth adapted mesh. Once again the velocity profiles from the triangular meshes get worse in comparison as we move further downstream from the trailing edge. This test case further confirms that the quad dominant advancing layer meshes can track the wake more efficiently.

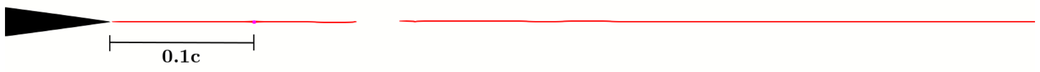
4.2.3 Viscous flow at $Re = 5000$, $Ma = 0.5$ and $\alpha = 0°$

The third adaptation case is $Re = 5000$, Mach number $Ma = 0.5$ and angle of attack $\alpha = 0°$. The initial mesh used was an isotropic unstructured mesh around NACA 0012 airfoil with 6654 cells. At these flow conditions, we get a steady separation in the wake but due to poor resolution of the initial mesh, the flow separation was not observed until the first adaptation. The wake centerline for the separated flow was tracked using three steps: (1) backward integration in time from the trailing edge, (2) find the saddle point, and (3) add a horizontal offset from saddle point to escape separation bubble and do forward integration in time. The flow separation observed is depicted in Figure 4.24a and Figure 4.24b shows the streamline obtained from streamline tracking algorithm. The streamline, in this case, is split into two curves and these curves were joined together to give the single curve depicted in Figure 4.24c so that the adapted mesh (which already has the wake centerline inserted into it) can be morphed to match the new wake centerline. The process was repeated for each cycle of mesh adaptation.

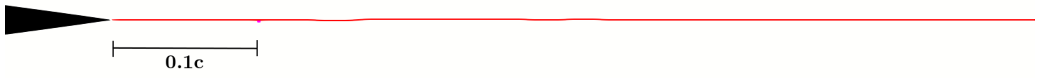
The position of the wake centerline on the initial and final mesh is depicted in Figure 4.25. The comparison shows that the wake centerline position from the final adapted mesh is more accurate since it coincides with the symmetry axis for this symmetric flow. The mesh morphing result for this case is shown in Figure 4.26.



(a) Paraview streamtracer result showing flow separation



(b) Streamline tracking algorithm result



(c) Streamline after joining two curves in Figure 4.24b

Figure 4.24: Wake centerline for separated flow

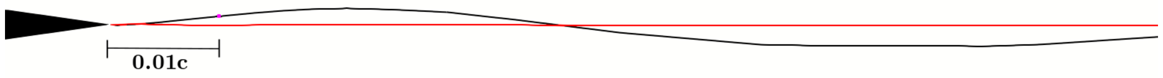


Figure 4.25: Wake centerline position - initial (black) and final (red) at $Re = 5000$, $Ma = 0.5$ and $\alpha = 0^{\circ}$ - Closeup view at the trailing edge

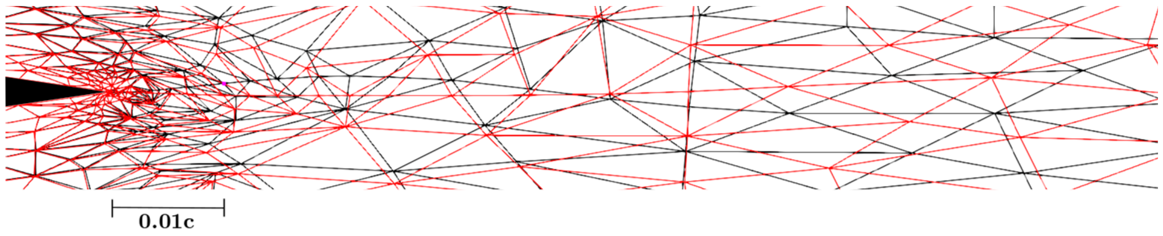
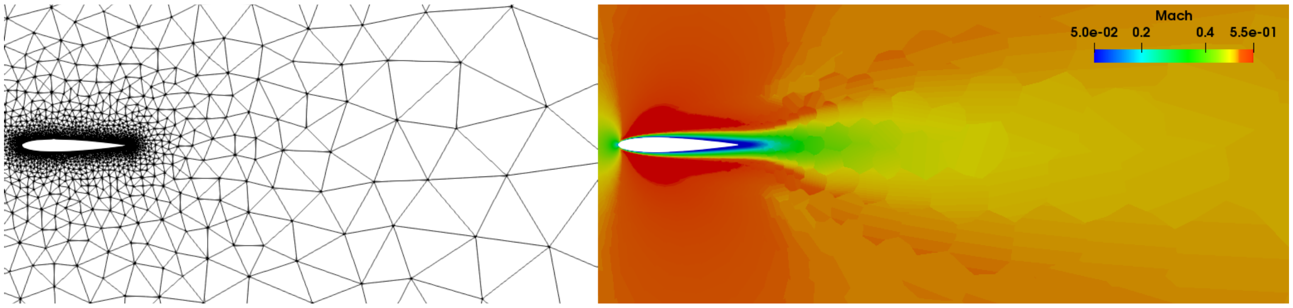
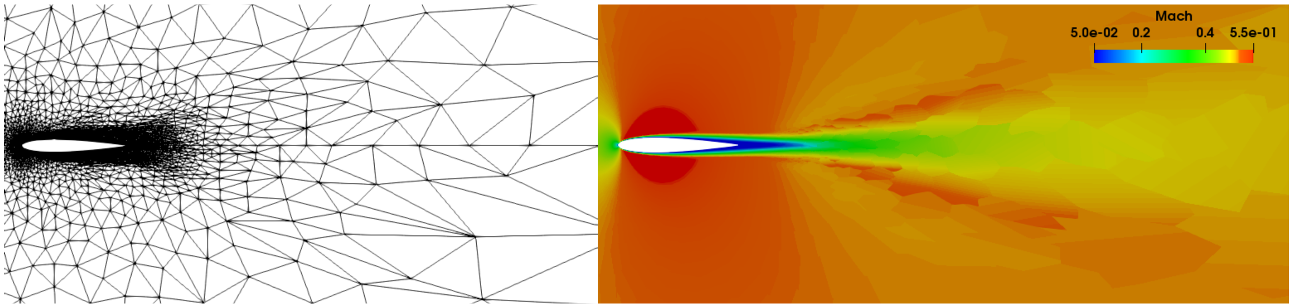


Figure 4.26: First adapted mesh (black) and morphed mesh (red) before second adaptation

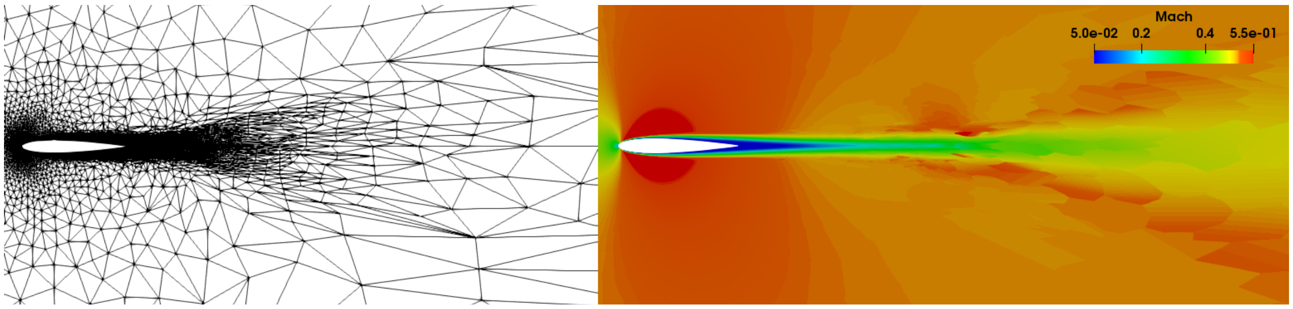
Figure 4.27 shows the solution on each mesh for this case and shows improvement in the resolution of the boundary layer and wake with each mesh refinement.



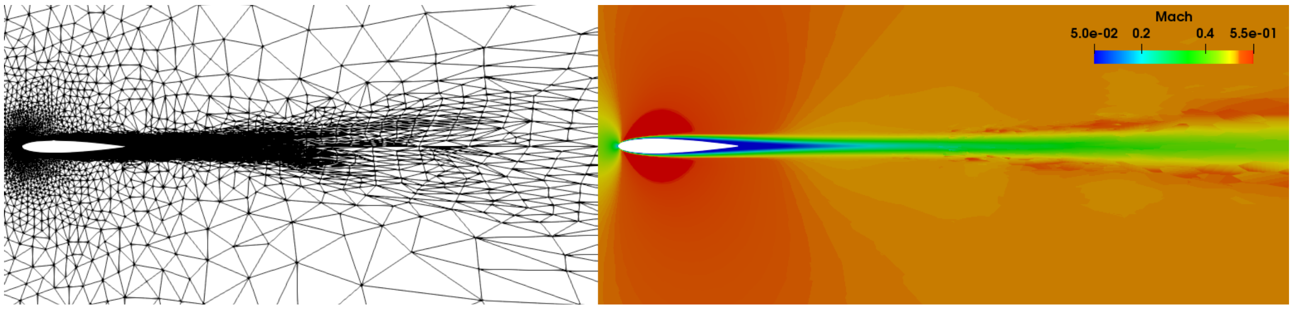
(a) Solution on initial mesh (3543 vertices)



(b) Solution after first adaptation (5515 vertices)



(c) Solution after second adaptation (8851 vertices)



(d) Solution on final mesh (12903 vertices)

Figure 4.27: Adapted meshes and flow solutions with wake centerline at $Re = 5000$, $Ma = 0.5$ and $\alpha = 0^{\circ}$

	Mesh	$nDOF$	C_L	C_D
Present results	Initial	3543	0.003618	0.064222
	1st adapted	5515	0.000360	0.056852
	2nd adapted	8851	0.000173	0.055826
	3rd adapted	12903	0.000155	0.055724
	Advancing layer	5006	0.000050	0.056000
	Advancing layer	6766	0.000040	0.056000
Zuniga Vazquez [55]	Initial	2543	0.014490	0.061180
	1st adapted	4235	0.016440	0.034844
	2nd adapted	6688	0.001424	0.054950
	3rd adapted	10335	0.004885	0.054430
Sharbatdar [49]	Initial	2543	0.015074	0.903403
	1st adapted	3899	0.009244	0.76383
	2nd adapted	9140	0.000203	0.056555
	3rd adapted	19964	0.000108	0.055230
4^{th}order Spectral Volume [22]		34560	-	0.054672
ARC2D (320×128) [16]		40960	-	0.054200
R. C. Swanson and S. Langer (4096×2048) [51]		8388608	-	0.055649

Table 4.2: Comparison of lift and drag coefficients for NACA 0012 airfoil at $Re = 5000$, $Ma = 0.5$ and $\alpha = 0^\circ$

Table 4.2 summarizes some of the published results for this test case. The lift coefficient in our scheme converges faster. For example, the second adapted mesh with wake centerline gives a smaller error in lift coefficient compared to the second adapted mesh of Sharbatdar [49] and third adapted mesh results from Zuniga Vazquez [55], both of which have more vertices. The drag coefficient on the final meshes is similar to these published results. However, our result of drag coefficient on the third adapted mesh is closer to results obtained on an extremely fine mesh by R. C. Swanson and S. Langer of NASA Langley Research Center [51].

Once again the advancing layer mesh was generated for this case based on the initial geometry shown in Figure 4.28. The flow solutions on final adapted triangular and advancing layer quad dominant meshes are depicted in Figure 4.29, showing that the advancing layer mesh can track the wake for longer distance with many fewer degrees of freedom.

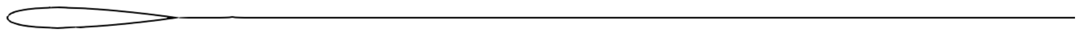
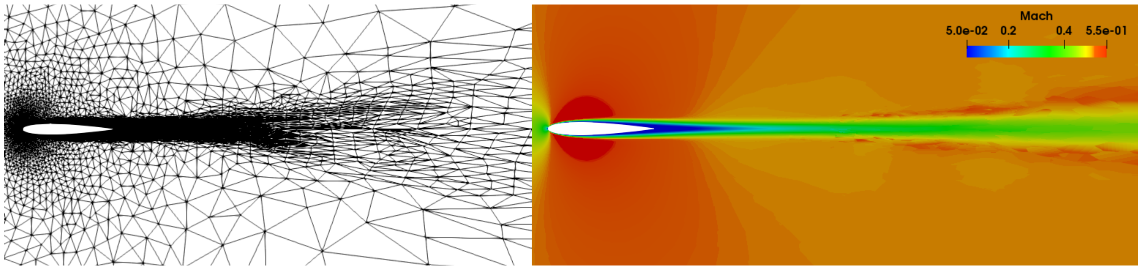
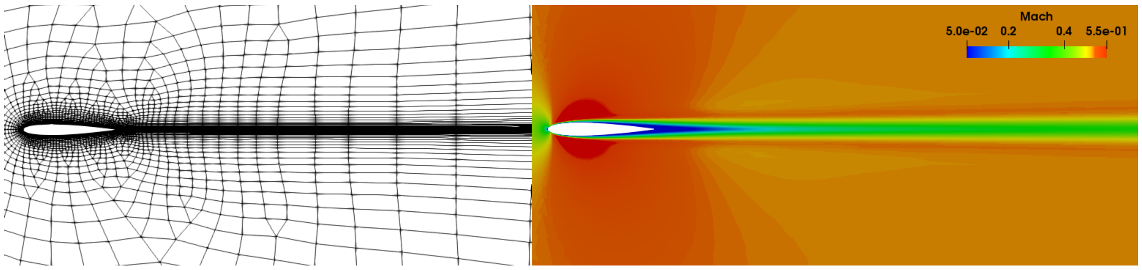


Figure 4.28: Initial geometry for quad dominant mesh at $Re = 5000$, $Ma = 0.5$ and $\alpha = 0^\circ$



(a) Solution on final adapted triangular mesh (12903 vertices)

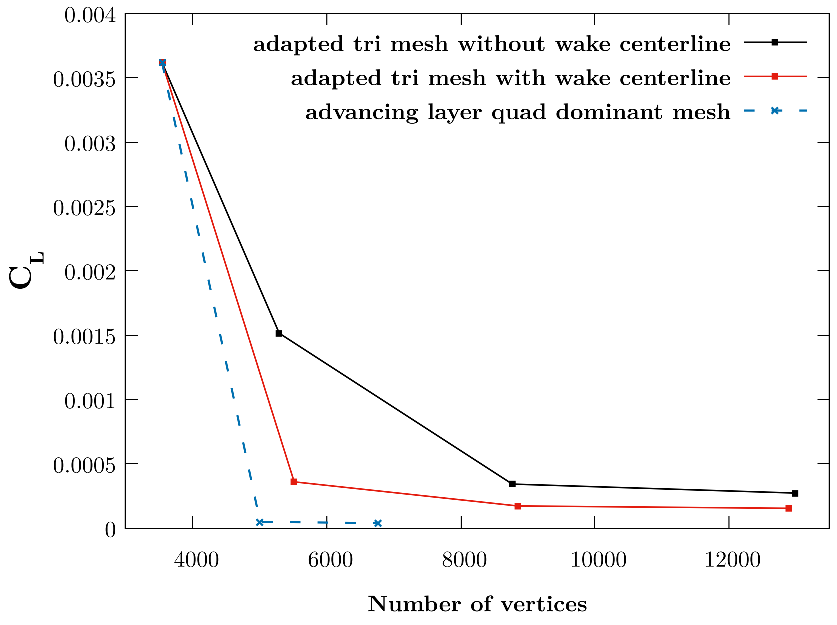


(b) Solution on advancing layer quad dominant mesh (5006 vertices)

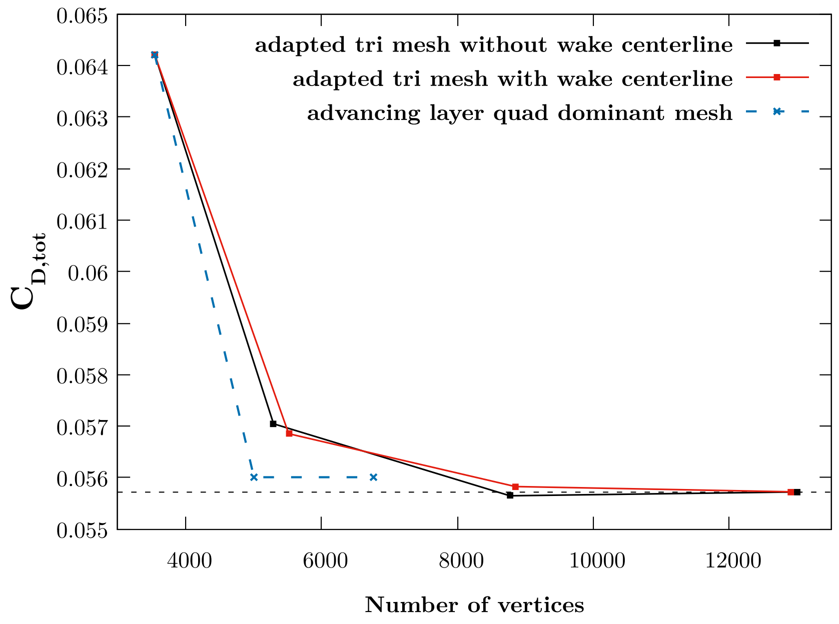
Figure 4.29: Flow solutions on final adapted triangular mesh and advancing layer mesh at $Re = 5000$, $Ma = 0.5$ and $\alpha = 0^{\circ}$, showing better solution behavior in the wake for the advancing layer mesh

The plots for the convergence history of aerodynamic coefficients against degrees of freedom are presented in Figure 4.30. The lift coefficient converges faster on advancing layer meshes and has smaller error compared to both triangular meshes since we expect to get $C_L = 0$ for this symmetric flow. However, comparing the triangular meshes only, the lift coefficient on mesh with wake centerline converges faster as it better symmetry compared to mesh without wake centerline. The convergence of the drag coefficient is roughly similar for triangular meshes but faster on the quad dominant advancing layer meshes.

Figure 4.31 shows the convergence history with respect to total CPU time for meshing and solver. For triangular meshes, the total CPU time for meshes with wake centerline is less compared to the meshes without the wake centerline. Nevertheless, the total CPU time for the advancing layer quad dominant meshes is much less compared to both the triangular meshes with advancing layer meshes giving a better result for lift coefficient and roughly same drag coefficient as grid converged value on final adapted triangular mesh. The difference between the grid converged value of drag coefficient from adapted triangular meshes and the advancing layer quad dominant meshes is only 0.49%.

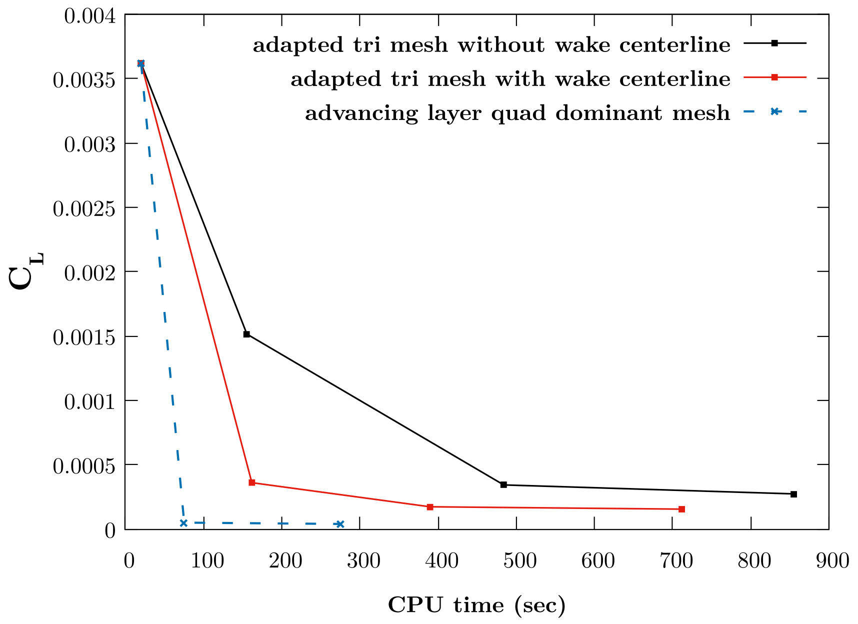


(a) Lift coefficient

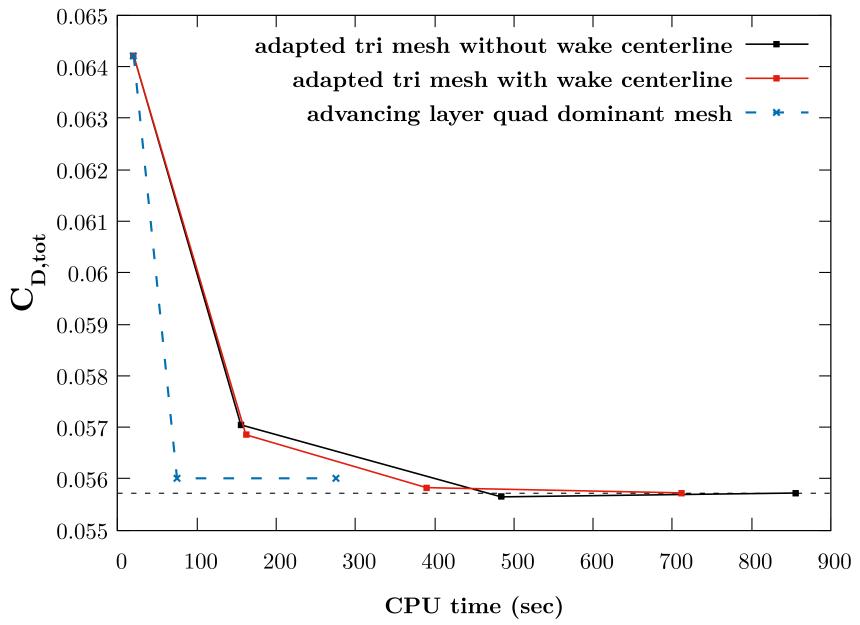


(b) Drag coefficient

Figure 4.30: Convergence of lift and drag coefficients against degrees of freedom (vertices) at $Re = 5000$, $Ma = 0.5$ and $\alpha = 0^\circ$

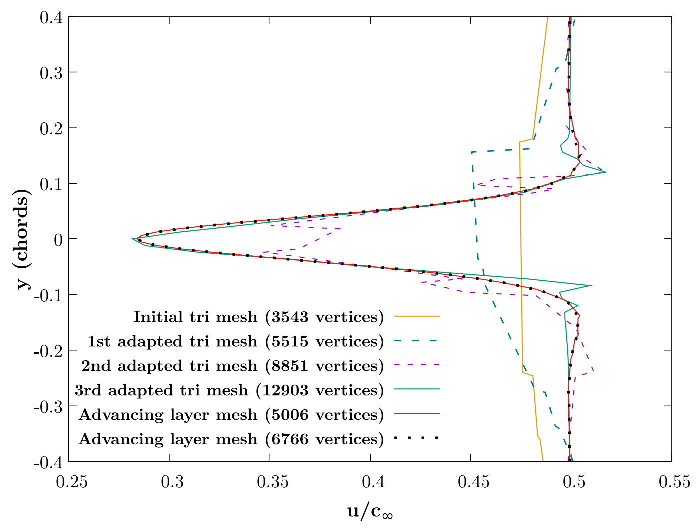


(a) Lift coefficient

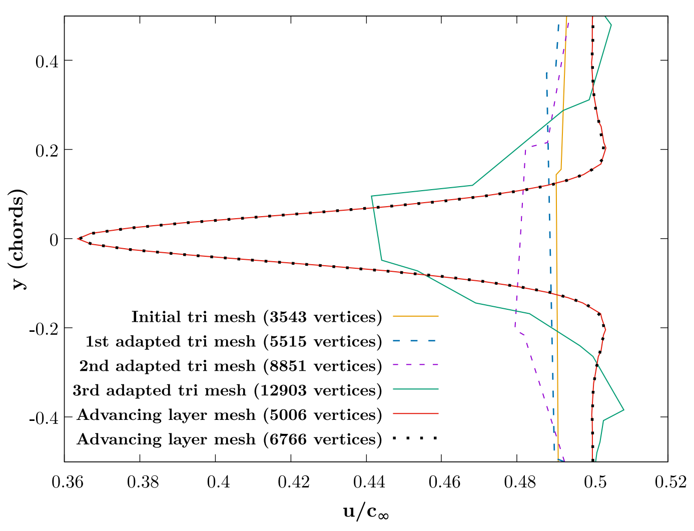


(b) Drag coefficient

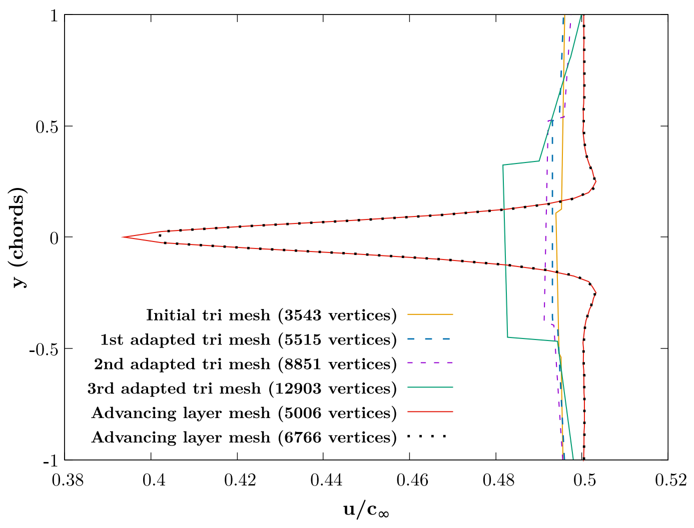
Figure 4.31: Convergence of lift and drag coefficients against total CPU time for meshing and solver at $Re = 5000$, $Ma = 0.5$ and $\alpha = 0^{\circ}$



(a) Two chords downstream of the trailing edge



(b) Five chords downstream of the trailing edge



(c) Eight chords downstream of the trailing edge

Figure 4.32: Velocity profiles in the wake for triangular and advancing layer quad dominant meshes

The comparison of velocity profiles in the wake between the advancing layer meshes and the adapted triangular meshes for this case is given in Figure 4.32. Once again, the advancing layer quad dominant meshes give the expected symmetrical velocity profile in the wake at different positions. The velocity profiles from advancing layer meshes with 5006 and 6766 vertices are identical at two and five chords downstream but slightly smoother for the more refined advancing layer mesh at eight chords downstream from the trailing edge. The velocity profile from the triangular meshes improves with each cycle of mesh adaptation but in this separated flow case, the adapted triangular meshes represent the wake velocity profile more poorly compared to previous test cases especially for five and eight chords downstream from the trailing edge where the wake velocity profile is poorly resolved even after the third adaptation. This comparison indicates that the advancing layer mesh tracks the wake better for separated flow as well.

4.3 Turbulent Subsonic Flow

Finally, we considered a case for turbulent subsonic flow. The free stream Mach number used was $Ma_\infty = 0.15$ while the Reynolds number based on the airfoil chord was $Re_c = 6 \times 10^6$. This test case is among of the validation case on NASA Turbulence Modeling Resource (TMR) website [48], which documents results from well known CFD codes. An unstructured hybrid mesh shown in Figure 4.34 was used. No-slip adiabatic boundary conditions were applied on the airfoil surface while the far-field boundary conditions (outflow/inflow based on the direction of velocity vector) were used at the outer boundary located $500c$ away from the airfoil. We also took into account the effect of curvature and used higher-order cubic cells for a more accurate representation of the boundary faces. However, streamline tracking in curved cells requires using the finite element mapping from straight cells to curved cells. We used the finite element mapping function implemented by Jalali [20] in ANSLib [40] and solved the non-linear system of equations using Newton's method to find the curved control volume containing the massless particle based on its position at a given time.

The system of equations solved using Newton's method to find the particle in curved cell was:

$$\begin{aligned} x &= \sum x_i \phi_i(\xi, \eta) \\ y &= \sum y_i \phi_i(\xi, \eta) \end{aligned} \tag{4.4}$$

where ϕ_i is the basis function at node i. We then solved for ξ and η such that:

$$\begin{aligned} x(\xi,\eta) &= x^* \\ y(\xi,\eta) &= y^* \end{aligned}$$

The procedure is as follow:

- First, find the straight cell in the straight mesh that contains the particle. This gives us a good initial guess.
- Then grab the neighbors around that cell and search them one by one. For each cell, solve for ξ and η in the reference element and ensure that ξ and η values are such that the point falls inside the triangle. Using the values of ξ and η, re-calculate the values of x and y. If x and y are the same as what we started with, then we are in the correct cell. Otherwise, we go to other neighbors and repeat the process until we find the desired cell.

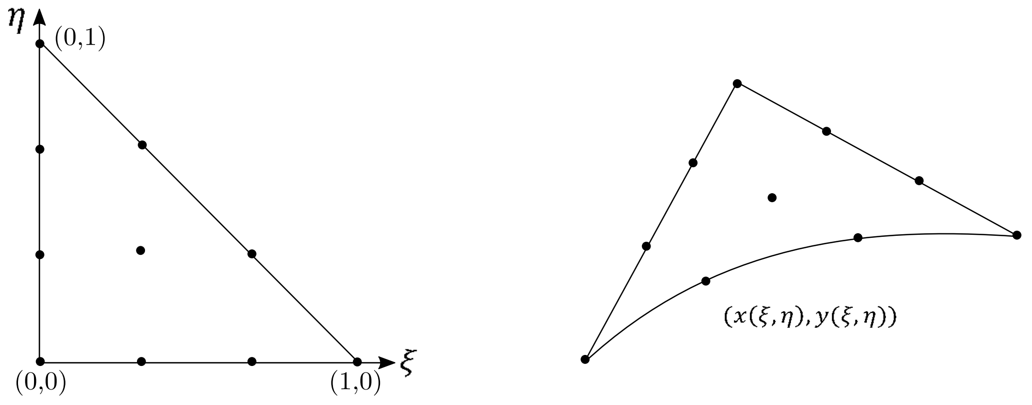


Figure 4.33: Mapping from reference triangle to cubic triangular cell

We computed the flow solution at different angles of attack, that is $\alpha = 0^{\circ}$, 4° and 10°, using the second-order accurate cell-centered finite volume solver and the velocity field was integrated to track the wake centerline in each case. Figure 4.35 shows the streamline tracking results. These results demonstrate the robustness of the streamline tracking algorithm and the ability to track the streamlines through mixed element meshes and high-order cubic curved cells. The flow, in this case, is attached and no flow separation was observed from $\alpha = 0^{\circ}$ to 10°.

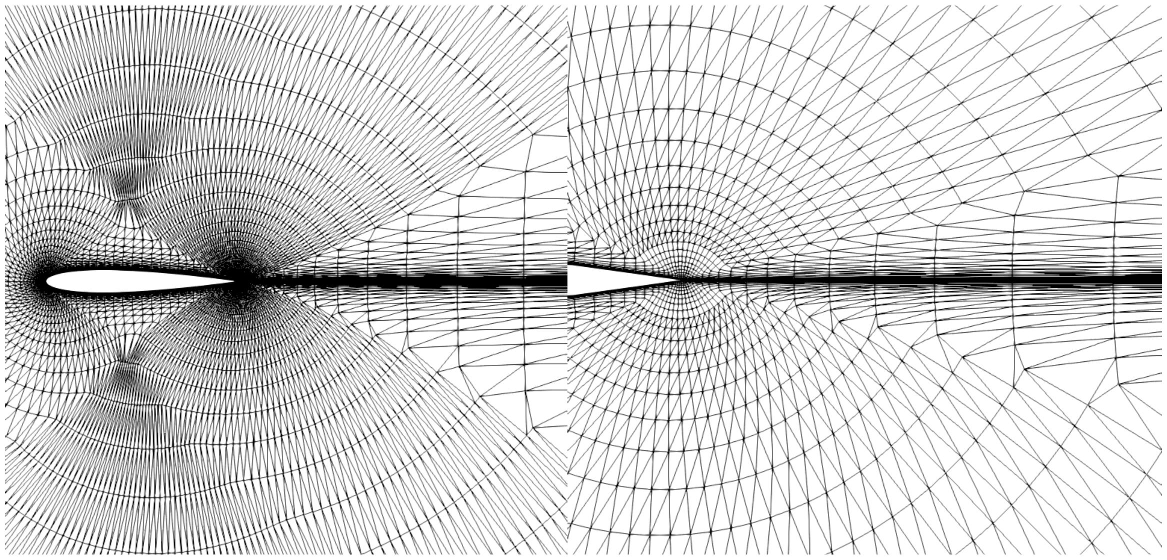


Figure 4.34: Unstructured hybrid mesh (n.DoF $= 25,088$)

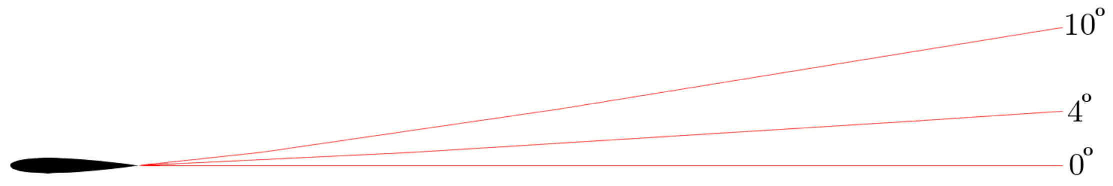


Figure 4.35: Wake centerlines at $Re = 6 \times 10^{6}$, $Ma_{\infty} = 0.15$ and $\alpha = 0^{\circ}$, 4°, 10° using the streamline-tracking algorithm

Chapter 5

Concluding Remarks

5.1 Summary

A new approach for mesh adaptation for wakes has been presented in this research in an effort to improve the solution accuracy and resolve the wake efficiently. The first step of this approach involved the development of streamline tracking algorithm in order to track the wake centerline. The velocity data was obtained from solution reconstruction using the second-order accurate vertex-centered finite volume solver and integrated numerically using the adaptive fourth-order Runge-Kutta method to obtain the wake centerline. The obtained streamline was then successfully inserted into the existing isotropic unstructured mesh using the surface insertion algorithm in GRUMMP. The surface insertion algorithm takes an initial unstructured mesh and geometry of wake centerline as two initial inputs and its main point is to recover the wake surface while not changing the mesh size at all more than a cell size or two from the wake surface.

In the next step, an existing anisotropic mesh adaptation scheme developed by Pagnutti [42] and further extended by Sharbatdar [49] and Zuniga Vazquez [56] was used. The metric based on solution approximation error was computed at each mesh vertex. The metric represents the desired anisotropy for each vertex and is a 2×2 symmetric positive definite matrix. The desired anisotropy was then communicated through the metric to the mesh adaptation code in GRUMMP to generate an anisotropic mesh. The mesh adaptation scheme produced highly anisotropic and quasi-structured cells where needed based on derivatives of flow solution variables. Unlike isotropic mesh adaptation which only provides mesh resolution in desired areas, anisotropic mesh adaptation also provides proper cell alignment with the solution in addition to mesh resolution to capture anisotropic flow features. The high aspect ratio of anisotropic cells and regular structure of quasi-structured cells allows to capture the flow information much more efficiently compared to isotropic cells. Refining the mesh in such a way based on solution features helps to produce an optimal mesh for computing the solution. The mesh adaptation scheme produces high quality meshes using the four principal operations for mesh quality improvement as discussed in Section 2.5. The mesh modification operations preserve the wake centerline and do not move the vertices on it at all. After obtaining the adapted mesh, the solution was re-computed and a new wake centerline was obtained.

In the second and subsequent mesh adaptation cycles, the mesh was morphed and mesh points from old wake centerline were projected to match the new wake centerline by solving the linear elasticity problem, using displacement between two wake centerlines as constraints. This helped to move the wake centerline to a more accurate position and preserved the already adapted regions, thus making the adaptation process more efficient.

An advancing layer mesh generator was used as an alternative technique to produce an anisotropic quad dominant mesh and the results are compared with triangular meshes with and without the wake centerline. We also presented the comparison of velocity profiles in the wake at different locations downstream from the trailing edge for the advancing layer quad dominant and adapted triangular meshes.

5.2 Conclusions

To verify the correctness of streamline tracking, results are presented for an analytical velocity field which confirmed the order of accuracy of the numerical method and the ability to track streamline through a mesh using the reconstructed velocity data. The accuracy of the mesh morphing approach is verified by using the method of manufactured solutions which proved that the linear elasticity solver is second-order accurate.

Test cases for viscous flow at different flow conditions have been considered and the robustness of the mesh adaptation scheme is demonstrated by improvement in solution accuracy and resolution of the boundary layer and wake as the adaptation process is repeated. The flow separation not observed on the initial triangular mesh in the third test case highlights the fact that it is immensely important to have an appropriate resolution of flow solution in the boundary layer and wake. The wake centerline for the first and third test cases should coincide with the symmetry axis; comparison on initial and final adapted triangular meshes show that wake centerline on the final mesh is closer to the symmetry line. Comparing just the triangular meshes, the convergence of aerodynamic coefficients demonstrated that there is a small CPU time advantage in case of meshes with wake centerline and solution converges slightly faster especially for the third test case. We expect to get lift coefficient equal to zero at zero angle of attack but this is not the case due to asymmetry in the mesh. The triangular meshes with wake centerline have better symmetry compared to meshes without wake centerline which plays an important role in faster convergence of the solution.

Nevertheless, the comparison of results between the triangular meshes and the advancing layer quad dominant meshes showed that the aerodynamic coefficients converge much faster on the advancing layer meshes in all the test cases. There is a smaller error in lift coefficient on the advancing layer meshes for symmetric flow cases compared to both triangular meshes since the advancing layer quad dominant meshes have better symmetry compared to triangular meshes. Also, there is a clearly noticeable CPU time advantage in using the advancing layer meshes. This is because it is much cheaper to generate the advancing layer mesh rather than doing mesh adaptation which saves CPU time required for meshing. The advancing layer mesh can also track the wake better with many fewer degrees of freedom which helps in reducing the CPU time for the solver.

The velocity profiles from the advancing layer meshes provide evidence that it can capture the wake more efficiently even up to several chords downstream from the trailing edge. The velocity profile on the initial triangular meshes is not physical in all the cases due to poor mesh resolution in the wake. The velocity profiles from triangular meshes improve with mesh adaptation and approach the velocity profiles represented by the quad dominant advancing layer meshes. The final adapted triangular meshes already have almost twice the degrees of freedom compared to the quad dominant advancing layer meshes. However, the velocity profiles from triangular meshes get worse in comparison as we move further downstream from the trailing edge such as at five and eight chords especially for the third test case with flow separation. This comparison study illustrates that the advancing layer mesh is more efficient in resolving the wake.

Moreover, the robustness of the streamline tracking algorithm is further demonstrated by tracking the wake centerline at different angles of attack for a fully turbulent flow. The turbulent flow case demonstrates our ability to track streamlines through mixed element meshes and high-order cubic curved cells.

5.3 Recommended Future Work

This thesis demonstrated a new approach for anisotropic mesh adaptation for wakes. There are several areas to which the overall methodology can be extended and improved further.

- The streamline tracking algorithm tracks the streamlines for 2-D flows. One possible addition in the long run can be to extend the algorithm to 3-D flows and trace a stream surface.

- In the future, it will be good to do the anisotropic mesh adaptation for turbulent flow case as well and follow the mesh adaptation loop discussed in Section 3.5. It will also be good to solve the drag polar for turbulent flow.

- For quad dominant advancing layer mesh, we produced the mesh by using an average of the laminar boundary layer and laminar far-wake profiles. One potential improvement for this can be to explore ways to get a better estimate of the overall profile that can resolve the wake even better especially in the near-wake region.

- In addition to resolving the boundary layer and wake, we also need to resolve the shock if the flow is transonic. Considering this additional feature of transonic flow, it will be good to identify the shock surface and insert it into the mesh along with the wake centerline and do anisotropic mesh adaptation. Also, then the initial geometry for the advancing layer approach can be modified to include both the wake centerline and the shock and march off both the surfaces to generate the advancing layer mesh. This might help to resolve the shock more efficiently in addition to the wake.

Bibliography

[1] National Aeronautics and Space Administration (NASA). Definition of streamlines. https://www.grc.nasa.gov/www/k-12/airplane/stream.html, May 2015. Accessed: 2017-12-30.

[2] Zaib Ali, Paul G Tucker, and Shahrokh Shahpar. Optimal mesh topology generation for CFD. *Computer Methods in Applied Mechanics and Engineering*, 317:431–457, 2017.

[3] 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, 1999.

[4] Thomas Apel. *Anisotropic Finite Elements: Local Estimates and Applications.* 1999.

[5] Stanley A Berger. *Laminar Wakes.* American Elsevier Publishing Company, Inc., New York, 1971.

[6] Carlo L Bottasso. Anisotropic mesh adaption by metric-driven optimization. *International Journal for Numerical Methods in Engineering*, 60(3):597–639, 2004.

[7] Michael L Brewer, Lori Freitag Diachin, Patrick M Knupp, Thomas Leurent, and Darryl J Melander. The Mesquite Mesh Quality Improvement Toolkit. In *Proceedings of the 12th International Meshing Roundtable,* pages 239-250, 2003.

[8] Zhiqiang Cai. On the finite volume element method. *Numerische Mathematik*, 58(1):713–735, 1990.

[9] Anil W Date. *Introduction to Computational Fluid Dynamics.* Cambridge University Press, 2005.

[10] D Davidson. The role of computational fluid dynamics in process industries. In *Frontiers of Engineering: Reports on Leading-Edge Engineering from the 2002 NAE Symposium on Frontiers of Engineering*, page 21. National Academies Press, 2003.

[11] MN Dhaubhadel. CFD applications in the automotive industry. *Journal of Fluids Engineering*, 118(4):647–653, 1996.

[12] Cécile Dobrzynski and Pascal Frey. Anisotropic delaunay mesh adaptation for unsteady simulations. In *Proceedings of the 17th International Meshing Roundtable*, pages 177–194. Springer, 2008.

[13] Lori A Freitag and Carl Ollivier-Gooch. The effect of mesh quality on solution efficiency. In *Proceedings of the Sixth International Meshing Roundtable*, page 249, 1997.

[14] Pascal-Jean Frey and Frédéric Alauzet. Anisotropic mesh adaptation for CFD computations. *Computer Methods in Applied Mechanics and Engineering*, 194:5068–5082, 2005.

[15] Gene H. Golub and Charles F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, 3rd edition, 1983.

[16] Carl F. Gooch. *Solution of the Navier-Stokes Equations on Locally-Refined Cartesian Meshes Using State-Vector Splitting.* PhD thesis, Stanford University, 1993.

[17] D Holmes and S Connell. Solution of the 2D Navier-Stokes equations on unstructured adaptive grids. In *9th Computational Fluid Dynamics Conference,* AIAA Paper 89-1932, 1989.

[18] Weizhang Huang. Metric tensors for anisotropic mesh generation. *Journal of Computational Physics*, 204(2):633–665, 2005.

[19] Florence V Hutcheson, Daniel Stead, and Gerald Plassman. Experimental study of wake/flap interaction noise and the reduction of flap side edge noise. In *22nd AIAA/CEAS Aeroacoustics Conference,* AIAA paper 2016-2955, 2016.

[20] Alireza Jalali. *An adaptive higher-order unstructured finite volume solver for turbulent compressible flows.* PhD thesis, The University of British Columbia, Department of Mechanical Engineering, 2017.

[21] Alireza Jalali and Carl F Ollivier Gooch. Higher-order finite volume solution reconstruction on highly anisotropic meshes. In *21st AIAA Computational Fluid Dynamics Conference,* AIAA Paper 2013-2565, 2013.

[22] Ravi Kannan and Zhijian Wang. Improving the high order spectral volume formulation using a diffusion regulator. *Communications in Computational Physics*, 12(1):247–260, 2012.

[23] Abhishek Khare, Ashish Singh, and Kishor Nokam. Best practices in grid generation for CFD applications using hypermesh. *Computational Research Laboratories*, page 2, 2009.

[24] P Knupp, L Freitag-Diachin, and B Tidwell. Mesquite: Mesh Quality Improvement Toolkit User's Guide. *Sandia National Laboratories*, 2013.

[25] Randall J LeVeque. *Finite Volume Methods for Hyperbolic Problems* (Cambridge Texts in Applied Mathematics). Cambridge University Press, 2002.

[26] Zhenquan Li and Gordon Mallinson. An adaptive streamline tracking method for two dimensional CFD velocity fields. In *Proceedings of the 7th Asian Symposium on Visualization*, pages 1–6, Singapore, November 2003.

[27] Vladimir D Liseikin. *A computational differential geometry approach to grid generation.* Springer Science & Business Media, 2006.

[28] Harvard Lomax, Thomas H Pulliam, and David W Zingg. *Fundamentals of Computational Fluid Dynamics*. Springer Science & Business Media, 2013.

[29] Adrien Loseille. Metric-orthogonal anisotropic mesh generation. *Procedia Engineering*, 82:403 – 415, 2014. 23rd International Meshing Roundtable (IMR23).

[30] Adrien Loseille and Rainald Lohner. Anisotropic adaptive simulations in aerodynamics. In *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, page 169, 2010.

[31] Shahzaib Malik and Carl F Ollivier Gooch. Mesh Adaptation for Wakes via Surface Insertion. In *AIAA SciTech 2019 Forum,* AIAA Paper 2019-1996, 2019.

[32] DJ Mavriplis. Unstructured grid techniques. *Annual Review of Fluid Mechanics*, 29(1):473–514, 1997.

[33] Fotis Mavriplis. CFD in aerospace in the new millennium. *Canadian Aeronautics and Space Journal*, 46(4):167–177, 2000.

[34] Krzysztof Michalak and Carl Ollivier-Gooch. Matrix-explicit GMRES for a higher-order accurate inviscid compressible flow solver. In *18th AIAA Computational Fluid Dynamics Conference,* AIAA Paper 2007-3943, 2007.

[35] F Moukalled, L Mangani, M Darwish, et al. *The finite volume method in computational fluid dynamics - An Advanced Introduction with OpenFOAM and Matlab.* Springer, 2016.

[36] 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):2582–2609, 2008.

[37] Logan Numerow and Carl F Ollivier Gooch. A mixed-element 3D advancing layer mesh generator for complex geometries. In *23rd AIAA/CFD Conference,* AIAA Paper 2017-3453, 2017.

[38] Carl Ollivier-Gooch. An unstructured mesh improvement toolkit with application to mesh improvement, generation and (de-) refinement. In *36th AIAA Aerospace Sciences Meeting and Exhibition,* AIAA Paper 98-0218, 1998. http://tetra.mech.ubc.ca/GRUMMP/.

[39] Carl 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.

[40] Carl F. Ollivier-Gooch. ANSLib: A scientific computing toolkit supporting rapid numerical solution of practically any PDE. In *Proceedings of the Eighth Annual Conference of the Computational Fluid Dynamics Society of Canada*, pages 21–28. Societe canadienne de CFD / CFD Society of Canada, June 2000.

[41] Doug Pagnutti. Anisotropic adaptation: Metrics and meshes. Master's thesis, The University of British Columbia, Department of Mechanical Engineering, 2008.

[42] Doug Pagnutti and Carl Ollivier-Gooch. A generalized framework for high order anisotropic mesh adaptation. *Computers and Structures*, 87(11-12):670–679, 2009.

[43] Michael A Park, Joshua A Krakos, Todd Michal, Adrien Loseille, and Juan J Alonso. Unstructured grid adaptation: Status, potential impacts, and recommended investments toward CFD Vision 2030. 2016.

[44] William H Press, Saul A Teukolsky, William T Vetterling, and Brian P Flannery. *Numerical Recipes 3rd edition: The Art of Scientific Computing.* Cambridge University Press, 2007.

[45] Shmuel Rippa. Long and thin triangles can be good for linear interpolation. *SIAM Journal on Numerical Analysis*, 29(1):257–270, 1992.

[46] Philip L Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357–372, 1981.

[47] Rolls-Royce. BR715 for the Boeing 717. https://www.rolls-royce.com/products-and-services/civil-aerospace/airlines/br715.aspx. Accessed: 2018-12-25.

[48] C Rumsey. Turbulence Modeling Resource, NASA Langley Research Center, 2018.

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

[50] Justin Solomon. *Numerical algorithms: methods for computer vision, machine learning, and graphics*. CRC Press, 2015.

[51] R Charles Swanson and S Langer. Comparison of NACA 0012 Laminar Flow Solutions: Structured and Unstructured Grid Methods. Technical Memorandum NASA-TM-2016-219003, NASA Langley Research Center, 2016.

[52] Joe F Thompson, Bharat K Soni, and Nigel P Weatherill. *Handbook of Grid Generation*. CRC press, 1998.

[53] W Tollmien. *Handbuch der Experimental Physik*. Vol. 4, Hydro- und Aero-Dynamik, Part 1. Akademie-Verlag, Leipzig, 1931.

[54] Jiyuan Tu, Guan Heng Yeoh, and Chaoqun Liu. *Computational Fluid Dynamics: A practical approach*. Butterworth-Heinemann, 2018.

[55] Zuniga Vazquez. Adaptive creation of orthogonal anisotropic triangular meshes using target matrices. Master's thesis, The University of British Columbia, Department of Mechanical Engineering, 2015.

[56] Zuniga Vazquez and Carl F Ollivier Gooch. Target-edge based orthogonal anisotropic mesh. In *22nd AIAA Computational Fluid Dynamics Conference,* AIAA Paper 2015-2294, 2015.

[57] Henk Kaarle Versteeg and Weeratunge Malalasekera. *An Introduction to Computational Fluid Dynamics: The Finite Volume Method*. Pearson Education, 2nd edition, 2007.

[58] Li Wang, W Kyle Anderson, and Jon Erwin. Solutions of high-order methods for three-dimensional compressible viscous flows. In *42nd AIAA Fluid Dynamics Conference and Exhibit,* AIAA Paper 2012-2836, 2012.

[59] Frank M White. *Viscous Fluid Flow.* McGraw-Hill Book Company, 3rd edition, 2006.

[60] Daniel W Zaide and Carl F Ollivier-Gooch. Inserting a surface into an existing unstructured mesh. *International Journal for Numerical Methods in Engineering*, 106(6):484–500, 2016.

Appendix A

Command-line Options

The command line options used to obtain the results in Chapter 4 are included below.

A.1 Wake Centerline Tracking

The executable file for wake centerline tracking in ANSLib is **streamline-tracking**. It can be executed through the Linux shell with the following command line options:

./streamline-tracking

Options

-f <mesh file>	Address of .mesh file without extension
-physics ()	RoeVisc2D for laminar and RoeTurbSA2D for turbulent flow
-mesh_type v	Vertex-centered mesh
-reynolds ()	Reynolds number
-mach ()	Mach number
-angle ()	Angle of attack
-initial_step	Initial step size (for forward and backward integration)
-CASE 2	CASE 2 is for actual velocity data from flow solver

Alternative Options

-initial_step_backward	Initial step size for backward integration only **(optional)**
-CASE ()	Use CASE 1 for analytical velocity field
-mesh_type c	Cell-centered mesh

A.2 Wake Centerline Insertion into the Mesh

The executable file in GRUMMP for two-dimensional curve insertion is **surfins2d**. The following command line options should be used:

./surfins2d

Options

-i <mesh file>	Address of .mesh file with extension
-z <boundary file>	Address of .bdry file without extension
-I	Melts internal boundary faces and gives .mesh.mesh file

A.3 Obtaining the metric file

The metric file can be obtained from ANSLib using the **Driver** executable. The command line options are:

./Driver

Options

-f <mesh file>	Address of .mesh file without extension
-physics RoeVisc2D	Physics
-mesh_type v	Vertex-centered mesh
-reynolds ()	Reynolds number
-mach ()	Mach number
-angle ()	Angle of attack
-AA	Anisotropic adaptation
-r 2	Adaptation order

A.4 Anisotropic Mesh Adaptation (Metric-Based)

The executable file used for metric-based anisotropic mesh adaptation is **tmop_adapt2d** and can be found in GRUMMP. The following command line options should be used:

./tmop_adapt2d

Options

-i <mesh file>	Address of .mesh file or .grmesh file with extension
-B <boundary file>	Address of .bdry file without extension
-I	Melts internal boundary faces if .grmesh file is used

Additional file that must be provided

A metric file must be provided for the corresponding mesh file and should have same extension as mesh file with additional .metric at the end. For example, if the mesh file is 0012.mesh, then the metric file should be 0012.mesh.metric.

Note

For .grmesh, the boundary file should have coordinates of the curve (e.g wake centerline) inserted into the mesh.

A.5 Mesh Morphing

The executable file used for mesh morphing is **LinearElements** and can be found in ANSLib. The following command line options should be used:

./LinearElements

Options for general cases

-i <mesh file>	Address of .mesh file without extension
-B <boundary file>	Address of .bdry file without extension (Initial wake centerline)
-C <boundary file>	Address of .bdry file without extension (New wake centerline)

Options to test mesh morphing code

-i <mesh file>	Address of .mesh file without extension
-MMS	Method of manufactured solutions

A.6 Advancing Layer Mesh Generation

The executable file used for advancing layer mesh generation is **edam2d** and can be found in GRUMMP. The following command line options should be used:

./edam2d

Options

-i <boundary file>	Address of .bdry file without extension
-o <output file name>	Name for output mesh file
-g ()	Off-wall growth ratio per layer
-R ()	Mesh resolution
-G ()	Mesh grading
-n ()	Total number of layer to extrude
-N ()	Number of layer across boundary layer/wake thickness
-S ()	Stretching factor across boundary layer/wake thickness
-q ()	Reynolds number based on chord