

DNS of Flow over a T106A Low Pressure Turbine Using a Discontinuous-Galerkin Spectral-Element Method.

by
Maria Ruiz
CID: 01264365
Supervisor: Professor Sherwin

Department of Aeronautics
Imperial College London

MSc in Advanced Computational Methods for Aeronautics,
Flow Management and Fluid Structure Interaction,
Imperial College London, 2016-2017

Abstract

The Spectral/HP element method has been applied to perform Direct Numerical Simulations (DNS) over a T106A turbine blade cascade at a Reynolds number of 60,000 using the open source software *Nektar++*. The main goal of the thesis is to investigate the potential of the Compressible Solver and to compare the performance with the Incompressible solver. First, compressibility effects were assessed under three different Mach numbers at a Reynolds number of 8,000. This was done by solving the compressible Navier-Stokes equations with the Compressible Flow Solver and the unsteady incompressible Navier-Stokes equations through the Velocity Correction Scheme with the Incompressible Solver and comparing the results. The pressure and friction coefficients along the blade and the boundary layer parameters were analysed. The compressibility effects were found to be relevant even at low Mach numbers. Secondly, the Reynolds number was increased to 60,000, maintaining a low Mach number to mitigate compressibility effects and the results obtained through both solvers were compared. The Compressible solver was found to have three main downsides compared to the Incompressible solver. The main shortcoming of the Compressible solver is the numerical stability. Global de-aliasing technique addressing geometrical and PDE aliasing was applied to enhance stability of the simulations. Artificial viscosity turned out to be essential to deal with flow discontinuities when initializing the flow. The stability performance of both solvers was assessed by comparing the maximum time step allowing for a stable simulation under six different flow cases. The second concern found was the discrepancy between the prescribed and actual inlet conditions for the Compressible solver. The variances were investigated for different inlet conditions. The last inconvenience when using the Compressible solver was the lack of information in the user-guide. This lead to the development of a tutorial for the compressible flow solver and the update of the user-guide. Lastly, a study of the scalability of the problem when running multiple processes was performed.

Acknowledgments

I would like thank Professor Spencer Sherwin, who guided and supported me throughout my master thesis. I have much appreciated his help and constructive suggestions during this project.

Special thanks to Andrea Cassinelli for his invaluable and constant help.

I would also like to thank Douglas Serson, Daniele de Grazia, David Moxey and Gianmarco Mengaldo for their technical advices and assistance.

Lastly, I would like to acknowledge my gratitude to my family and friends for their encouragement throughout this year.

Contents

1 Introduction	1
1.1 DNS of Low-Pressure Turbine Cascade	1
1.2 Motivation and Objectives	2
1.3 Report Structure	2
2 Numerical Methods	3
2.1 Governing equations	3
2.2 Numerical approximations of PDEs.	4
2.3 Discontinuous Spectral/hp element methods.	5
2.4 Numerical Discretisation for first order problems	6
2.5 Numerical Discretisation for second order problems	7
2.6 Riemann Solvers	8
2.7 Implementation of the Boundary Conditions	9
2.8 De-Aliasing Strategies	10
2.9 Non-Smooth Artificial Viscosity	11
3 Problem Formulation	12
3.1 Flow Geometry and Mesh	12
3.2 Boundary Conditions	15
3.3 Initial Conditions	16
3.4 Solver Configuration	17
4 Results	21
4.4 Compressibility Effects	21
4.5 DNS of Flow Over a T106a Turbine at Reynolds 60,000	28
4.6 Time Step Restriction	31
4.7 Inlet Boundary Condition Comparison	32
4.8 Scaling of the Parallel Simulation	33
5 Conclusion and Future Work	35
5.1 User Focused Conclusions and Future Work	35
5.1 Code Development Conclusions and Future Work	36
5 Bibliography	38

List of Figures

- 2.1 Discontinuous Spectral/hp element methods
- 3.1 T106A low-pressure blade cascade specifications from Stieger
- 3.2 Complete view of the computational domain and mesh
- 3.3 Boundary layer of the mesh
- 3.4 DNS Spatial Resolution in x for Reynolds number of 8,000
- 3.5 DNS Spatial Resolution in y for Reynolds number of 8,000
- 3.6 DNS Spatial Resolution in x and y for Reynolds number of 60,000
- 3.7 Instantaneous density waves created when initializing the flow.
- 3.8 Kinetic Energy Evolution
- 3.9 Time averaged pressure coefficient with varying averaging time at Reynolds 60,000
- 3.10 Time averaged friction coefficient with varying averaging time at Reynolds 60,000
- 4.1 Averaged pressure coefficient along the surface of a T106A blade at Reynolds 8,000
- 4.2 Averaged pressure coefficient along the surface of a T106A blade at Reynolds 8,000
- 4.3 Averaged friction coefficient along the surface of a T106A blade at Reynolds 8,000
- 4.4 Averaged friction coefficient along the surface of a T106A blade at Reynolds 8,000
- 4.5 Velocity Profile Comparison for Reynolds number 8,000
- 4.6 Velocity Profile Comparison for Reynolds number 8,000 and Mach 0.1
- 4.7 Velocity Profile Comparison for Reynolds number 8,000 and Mach 0.15
- 4.8 Velocity Profile Comparison for Reynolds number 8,000 and Mach 0.2
- 4.9 Boundary layer parameters comparison at Reynolds 8,000 and Mach 0.1
- 4.10 Boundary layer parameters comparison at Reynolds 8,000 and Mach 0.15
- 4.11 Boundary layer parameters comparison at Reynolds 8,000 and Mach 0.2
- 4.12 Time Averaged quantities for Reynolds number 60,000 at Mach 0.1
- 4.13 Averaged pressure coefficient along the surface of a T106A blade at Reynolds 60,000
- 4.14 Averaged friction coefficient along the surface of a T106A blade at Reynolds 60,000
- 4.15 Velocity Profile Comparison for Reynolds number 60,000 and Mach 0.1

- 4.16** Displacement and momentum thickness boundary comparison between solvers at Reynolds 60,000 and Mach 0.1
- 4.17** Shape parameter comparison between solvers at Reynolds 60,000 and Mach 0.1.
- 4.18** Time Step Limitation Comparison between solvers
- 4.19** Inlet Conditions Discrepancy for the Reynolds number 8,000.
- 4.20** Computational time per time-step against the number of CPUs.
- 4.21** Speed-up against the number of CPUs.
- 4.22** Efficiency against the number of CPUs.

Chapter 1

Introduction

In this chapter the importance of the case study is presented with an overview of the software and method used for its resolution. Successively, the motivations and the objectives of the thesis are detailed. In the last section, the report structure is outlined.

1.1 DNS of Low-Pressure Turbine Cascade

Gas turbines plays a significant role in the global economy, being widely used in different industries. Convectional power plants, nuclear power stations, aero-engines and shipping are among the main applications of gas turbines. It is, hence, obvious that any increase in the efficiency of gas turbines would have a great impact, which explains the significant effort toward improving their design [1]. One major development is the increment of the reliability and accuracy of Computational Fluid Dynamics (CFD), that are recognized to be of extreme importance for the design.

Turbomachinery flow involves both complex physics, such as waves impingement upon blades and aeroacoustics phenomena, and complex moving geometries to resolve the tight clearances between components, which ultimately makes it a challenge for CFD. In recent years, with the success of High Performance Computing, Direct Numerical Simulations (DNS) has become a powerful tool. This computational expensive numerical technique offers more accurate and better resolved simulations, giving solution to complicated flow phenomena and opening the door to a new way of advancement of turbomachinery design [2]. There are multiple studies recently reported applying DNS technique to turbomachinery applications [3] [4] [5] [6].

In the past few decades, spectral/*hp* element methods have gained significant attention in industry and academia. These numerical methods divide the domain in separate elements, applying a polynomial expansion of arbitrary order P to each element. The advantages being the arbitrary order of spatial accuracy while maintaining a relatively low computational cost and the geometric flexibility they present. The combination of these two assets makes the Spectral/*hp* element advantageous for Direct Numerical Simulations where high spatial resolution is required, and therefore a potential tool for turbomachinery design [7].

In this thesis, the Spectral/*hp* element method is employed to perform DNS of flow in a low-pressure T106A turbine airfoil cascade using *Nektar++*. This open-source software provides high-performance scalable solvers for partial differential equations using the Spectral/*hp* element

method. Among the several solver applications *Nektar++* supports, this thesis focuses on the Compressible Flow and Incompressible Navier-Stokes solvers [8].

1.2 Motivation and Objectives

The main objective is to conduct numerical simulations on a T106A turbine blade cascade at a Reynolds number of 60,000 by using the Compressible and Incompressible Flow solvers and compare the results. In addition, the thesis gives a quantitative answer to how important compressible effects are for the T106A turbine blade cascade case. This was done by solving the unsteady compressible and incompressible Navier-Stokes equations under different Mach number cases and comparing the pressure and friction coefficient and analysing the boundary layer. Moreover, the performance of the Compressible and Incompressible solver is compared in terms of numerical stability. Also, a scalability analysis is performed. Lastly, this project aims to facilitate the user to run stable simulations with the Compressible Flow solver. This was achieved by creating a tutorial describing the main features in a simple manner and by updating the user-guide to include de-aliasing techniques and artificial viscosity strategy.

1.3 Report Structure

This report is divided into five chapters. In the first chapter the subject matter was introduced and the main motivations and objectives of the thesis were set. In chapter 2, the theoretical foundation of the numerical methods used in this thesis are presented. This includes, the description of the governing equations, the most common spatial discretisations for solving partial differential equations, the Spectral/hp element method, the two discontinuous spatial discretisations used, namely the DG method and the FR approach, the implementation of the boundary conditions, the De-Aliasing technique, the Riemann solvers and the non-smooth artificial viscosity. In chapter 3, the problem formulation is described. This section details the flow geometry and mesh used, the boundary and initial conditions and the solver configuration. Chapter 4 contains the results of the Direct Numerical Simulations. Within this chapter, the results and time step restrictions obtained with the Compressible and Incompressible solver under the same flow conditions are compared. Also, a study of the scalability is performed. Finally, chapter 5 includes the conclusions and recommendations for future work. This thesis involved the development of a tutorial and the update of the user-guide for the Compressible Flow Solver, presented in the appendices.

2.1 Governing Equations

The governing equations considered in this work are the 2D compressible Navier-Stokes equations. The second-order partial differential equations in a two-dimensional physical domain Ω and in a Cartesian frame of reference are written as:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_1}{\partial x} + \frac{\partial \mathbf{f}_2}{\partial y} = 0, \quad (2.1)$$

where $\mathbf{u}$ is the vector of the conserved variables and $\mathbf{f}_1 = \mathbf{f}_1(\mathbf{u}, \nabla_x \mathbf{u})$, $\mathbf{f}_2 = \mathbf{f}_2(\mathbf{u}, \nabla_x \mathbf{u})$ are the vector of fluxes that can be expressed in terms of its inviscid and viscous fluxes:

$$\mathbf{u} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix}, \quad \mathbf{f}_1 = \mathbf{f}_{i,1} - \mathbf{f}_{v,1}, \quad \mathbf{f}_2 = \mathbf{f}_{i,2} - \mathbf{f}_{v,2}. \quad (2.2)$$

where ρ is the density, u and v are the velocity components in x and y directions and E is the total energy. In this work, we considered a perfect gas law for which the pressure p is related to the total energy by the following expression:

$$E = \frac{p}{\gamma - 1} + \frac{1}{2} \rho (u^2 + v^2), \quad (2.3)$$

where γ corresponds to the ratio of specific heats.

The inviscid fluxes $\mathbf{f}_{i,1}$ and $\mathbf{f}_{i,2}$ from equation 2.2 are written as:

$$\mathbf{f}_{i,1} = \begin{pmatrix} \rho u \\ p + \rho u^2 \\ \rho uv \\ u(E + p) \end{pmatrix}, \quad \mathbf{f}_{i,2} = \begin{pmatrix} \rho v \\ \rho uv \\ p + \rho v^2 \\ v(E + p) \end{pmatrix}, \quad (2.4)$$

while the viscous fluxes $\mathbf{f}_{v,1}$ and $\mathbf{f}_{v,2}$ assume the following form:

$$\mathbf{f}_{v,1} = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{yx} \\ u\tau_{xx} + v\tau_{yx} + kT_x \end{pmatrix}, \quad \mathbf{f}_{v,2} = \begin{pmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ u\tau_{xy} + v\tau_{yy} + kT_y \end{pmatrix}, \quad (2.5)$$

where τ_{xx} , τ_{xy} , τ_{yx} and τ_{yy} are the components of the stress tensor¹

$$\begin{aligned} \tau_{xx} &= 2\mu \left(u_x - \frac{u_x + v_y}{3} \right), \\ \tau_{yy} &= 2\mu \left(v_y - \frac{u_x + v_y}{3} \right), \\ \tau_{xy} &= \tau_{yx} = \mu (v_x + u_y), \end{aligned} \quad (2.6)$$

being μ the dynamic viscosity according to the Sutherland's law and k the thermal conductivity.

¹ Note that we use Stokes hypothesis $\lambda = -2/3$

2.2 Numerical Approximations of PDEs

The approximation of an exact solution of a partial differential equation involves the discretization of the equations to satisfy a finite number of prescribed conditions. The definition of the conditions determines the numerical method or projection operator employed. The method of weighted residuals shows how different weight functions can be used to give rise to the most popular numerical methods for solving PDEs.

Suppose we have a linear partial differential equation defined over the domain Ω of the form:

$$\mathbb{L}(u) = 0, \quad (2.7)$$

subject to appropriate initial and boundary conditions. It is assumed that the solution $u(\mathbf{x}, t)$, can be accurately approximated by a linear combination of basis functions:

$$u(\mathbf{x}, t) \cong u^\delta(\mathbf{x}, t) = u_0(\mathbf{x}, t) + \sum_{i=1}^{N_{dof}} \hat{u}_i(t) \phi_i(\mathbf{x}), \quad (2.8)$$

where $u_0(\mathbf{x}, t)$ satisfies the initial and boundary conditions, $\hat{u}_i(t)$ are the N_{dof} unknown coefficients and $\phi_i(\mathbf{x})$ are the trial (or expansion) functions satisfying the homogeneous boundary conditions. Substituting the approximation into the differential operator a non-zero residual, R , is generated:

$$\mathbb{L}(u^\delta) = R(u^\delta). \quad (2.9)$$

In order to solve equation 2.9 a restriction is placed on the residual. This condition is usually applied by forcing to zero the Legendre inner product of the residual with respect to a weight or test function $v_j(\mathbf{x})$:

$$(v_j(\mathbf{x}), R) = \int_{\Omega} v_j(\mathbf{x}) R(\mathbf{x}) d\mathbf{x} = 0, \quad j = 1, 2, \dots, N_{dof} \quad (2.10)$$

This reduces to a system of ordinary differential equations in $\hat{u}_i(t)$, and for a time independent case, the result is a set of N_{dof} algebraic equations for the unknown constants $\hat{u}_i(t)$. Table 2.1 illustrates the different methods encompassed by the method of weighted residuals depending on the choices of the test functions $v_j(\mathbf{x})$ and the expansion functions $\phi_i(\mathbf{x})$.

Type of method	Test or Weight function
Collocation Method	$v_j(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_j)$
Finite Volume/Subdomain Methods	$v_j(\mathbf{x}) = \begin{cases} 1 & \text{inside } \Omega^j \\ 0 & \text{inside } \Omega^j \end{cases}$
Least-Squares Method	$v_j(\mathbf{x}) = \frac{\partial R}{\partial \hat{u}_j}$
Galerkin Method	$v_j(\mathbf{x}) = \phi_i$
Petrov-Galerkin	$v_j(\mathbf{x}) = \psi_i (\neq \phi_i)$

Table 2.1 Different computational method defined by the test functions $v_j(\mathbf{x})$ used [7].

However, the type of trial functions or approximation space are not defined. Spectral methods are characterised by using a set of global expansion functions, which are non-zero across the solution domain. On the other hand, the finite element methods use a set of expansion functions which are only defined in a local finite region. In *Nektar++* we focus on spectral/hp element methods, combining the local nature of the expansion functions typical of finite element methods and the arbitrary expansion functions common in spectral methods [7].

2.3 Discontinuous Spectral/hp element methods

The Spectral/hp element methods combines the partitioning of the computational domain into separate elements (h-type refinement) with spectral-like resolution properties (p-type refinement) in each element. Convergence is achieved both through mesh refinement or by increasing the polynomial order of the approximation. The spectral/hp can be described as a high-order finite element method, which integrates the high convergence and accuracy of the spectral methods and the geometric flexibility of the finite element method [7].

The construction of the spectral/hp element method involves the domain decomposition into non-overlapping elements. In order to facilitate the implementation of the main operations, each local element is mapped to a reference (or standard) element. On each standard element, the solution is represented through an arbitrary-degree expansion basis, and the operations required are performed. After all the multiplications, differentiations and integrations are implemented, the global solution is constructed via connectivity rules at the interface between adjacent elements. The original PDE is reduced to an algebraic system of equations, in the case of a time independent problem, or to a system of ODEs which requires to be advanced in time.

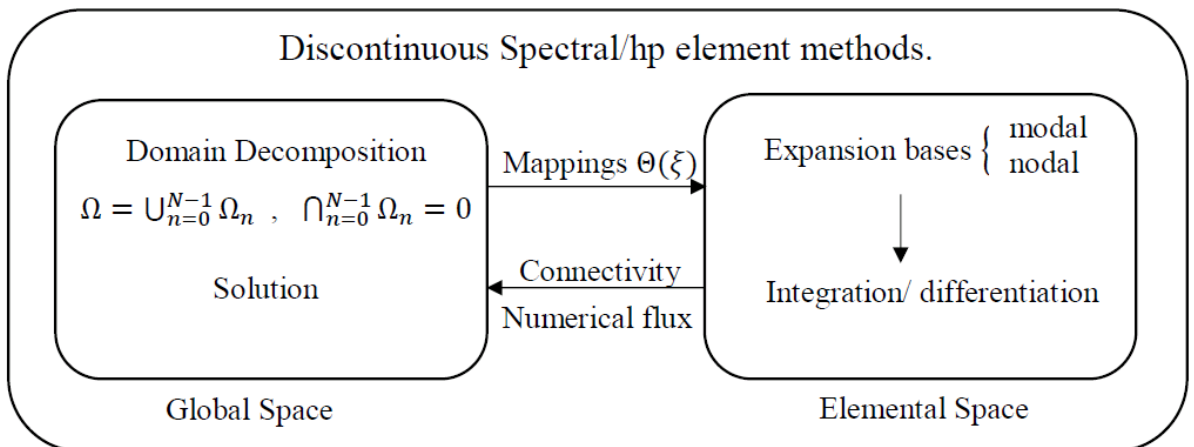


Figure 2.1: Discontinuous Spectral/hp element methods.

This work is concerned with discontinuous spectral/hp methods, in which we require the flux to be continuous between two adjacent elements rather than the solution. The connectivity

rules based on the physics of the problem are achieved through a boundary term composed by a numerical flux. The numerical flux $\mathbf{f}^{\delta I}(u_+^\delta, u_-^\delta)$ is obtained from the value of the solution on the left u_+^δ and right u_-^δ elements with respect to the interface. When discretizing a first order flux, namely the advective flux of the compressible Navier-Stokes equations, the compressible solver uses either approximated or exact Riemann solvers. In the case of second order flux, specifically, the viscous flux of the compressible Navier-Stokes, connectivity is also made through a numerical flux but of a slightly different form than the advection term [9].

In the compressible solver in *Nektar++* two different discontinuous spatial discretisation approaches are used, the discontinuous Galerkin (DG) and the flux reconstruction (FR) for first and second order problems.

2.4 Numerical Discretisation for first order problems

The Euler equations are a subset of the compressible Navier-Stokes equations. They consist in a first-order hyperbolic system of equations describing inviscid flows without thermal conductivity. The discretization of a first order problem by means of the DG and FR approaches involves dividing the domain Ω into N non-overlapping subdomains Ω_n and representing the solution by a polynomial of degree P within each element. The global solution is comprised by the summation of each elemental contribution [10].

2.4.1 Discontinuous Galerkin approach

The DG scheme takes into account the variational formulation of the equations, and when applied to the two-dimensional Euler equations the matrix form reads:

$$\frac{d\hat{\mathbf{u}}_n^\delta}{dt} = \mathbf{M}_n^{-1} \{ \mathbf{S}^\xi \hat{\mathbf{f}}_{i,1,n}^{\delta D} + \mathbf{S}^\eta \hat{\mathbf{f}}_{i,2,n}^{\delta D} - \tilde{\mathbf{b}}^{DG} \}, \quad (2.11)$$

where the superscript “D” indicates discontinuous quantities, $\mathbf{M}_n$ is the mass matrix, $\mathbf{S}^\xi$ and $\mathbf{S}^\eta$ are the advection matrices, $\hat{\mathbf{f}}_{i,1,n}$ and $\hat{\mathbf{f}}_{i,2,n}$ are the elemental nonlinear fluxes and $\tilde{\mathbf{b}}^{DG}$ is the surface integral:

$$\tilde{\mathbf{b}}^{DG} = \int_{\delta\Omega_e} \ell_{ij} (\tilde{\mathcal{H}}_i^{\delta I} \cdot \mathbf{n}^e) ds, \quad (2.12)$$

“with ℓ_{ij} being a two-dimensional polynomial expansion basis, $\mathbf{n}^e$ being the outward pointing normal of the face e of a given element, and $\tilde{\mathcal{H}}_i^{\delta I}$ being the tensor of the inviscid fluxes $[\hat{\mathbf{f}}_{i,1,n}^{\delta I}, \hat{\mathbf{f}}_{i,2,n}^{\delta I}]$ at the interface between two adjacent elements” [10].

2.4.2 Flux Reconstruction approach

The FR approach considers the differential form of the equations and when applied to the two-dimensional Euler equations, the matrix form reads:

$$\frac{d\hat{\mathbf{u}}_n^\delta}{dt} = \{\mathbf{D}_\xi \hat{\mathbf{f}}_{i,1,n}^{\delta D} + \mathbf{D}_\eta \hat{\mathbf{f}}_{i,2,n}^{\delta D} - \tilde{\mathbf{b}}^{FR}\}, \quad (2.13)$$

where $\mathbf{D}_\xi$, $\mathbf{D}_\eta$ are the differentiation matrices with respect to ξ and η , $\hat{\mathbf{f}}_{i,1,n}$ and $\hat{\mathbf{f}}_{i,2,n}$ are the elemental nonlinear fluxes and $\tilde{\mathbf{b}}^{FR}$ is the boundary term:

$$\tilde{\mathbf{b}}^{FR} = (\tilde{\mathcal{H}}_i^{\delta C} \cdot \mathbf{n}^e) \Psi' = \left[(\tilde{\mathcal{H}}_i^{\delta I} \cdot \mathbf{n}^e) - (\tilde{\mathcal{H}}_i^{\delta DI} \cdot \mathbf{n}^e) \right] \Psi', \quad (2.14)$$

“with Ψ' being the derivative of the correction function, $\tilde{\mathcal{H}}_i^{\delta I}$ being the tensor of the numerical inviscid fluxes $[\hat{\mathbf{f}}_{i,1,n}^{\delta I}, \hat{\mathbf{f}}_{i,2,n}^{\delta I}]$ at the interface between two adjacent elements and $\tilde{\mathcal{H}}_i^{\delta DI}$ being the tensor of the inviscid fluxes evaluated at the boundary of a given element” [10].

The final form of the DG and FR method can then be advanced in time via any explicit time discretisation scheme. In order to generate a stable numerical scheme, the coupling needs to be based on the natural propagation of the information. For the advection term of the compressible Navier-Stokes equations, the boundary term $\mathcal{H}_i^{\delta I}$ is calculated via a Riemann Solver. An intermediate flux at the interface between two adjacent elements is calculated by using characteristic information coming from an eigenvalue analysis of the equations and may contain information from the left (+) or right (-) faces of a given interface:

$$\tilde{\mathcal{H}}_i^{\delta I} = \tilde{\mathcal{H}}_i^{\delta I}(\hat{\mathbf{u}}_+, \hat{\mathbf{u}}_-). \quad (2.15)$$

In this work, we consider a weak implementation of the boundary conditions. Consequently, the term $\tilde{\mathcal{H}}^{\delta I}$ is responsible for the boundary conditions to be correctly transferred into the domain via a ghost state or direct calculation of the boundary term using the known BCs [10].

2.5 Numerical Discretisation for second order problems

The compressible Navier-Stokes equations consider fluid viscosity and heat conduction and consequently are composed by an inviscid and viscous tensor. They are function of the conserved variables and of their gradient, and are therefore a second-order system of equations. The numerical discretisation of inviscid fluxes are performed as shown in section 2.4 for the Euler equations. However, the viscous fluxes are treated differently. Equation 2.1 is split into two separate first order equations, an auxiliary equation and the principal equation formed by the original problem. To compute the derivatives of the viscous fluxes one need to perform

firstly the calculation of the spatial derivatives of the auxiliary variable, and to use these to calculate the viscous fluxes. For the DG and FR the numerical interface fluxes are computed though the local discontinuous Galerkin (LDG) method as follows:

$$\mathbf{u}_{aux}^{\delta I} = \{\mathbf{u}_{aux}^{\delta}\} \mp \beta [\mathbf{u}_{aux}^{\delta}], \quad (2.16)$$

$$\mathcal{H}_v^{\delta I} = \{\mathcal{H}_v^{\delta}\} \mp \beta [\tilde{\mathcal{H}}_v^{\delta}] - \gamma [\mathbf{u}_{aux}^{\delta}], \quad (2.17)$$

where

$$\{\mathbf{g}\} = \frac{\mathbf{g}_+ + \mathbf{g}_-}{2}, \quad [\mathbf{g}] = \frac{\mathbf{g}_+ \mathbf{n}_+ + \mathbf{g}_- \mathbf{n}_-}{2}, \quad \beta = \frac{1}{2} \mathbf{n}_+. \quad (2.18)$$

“with the quantities $\mathbf{g}_-$ and $\mathbf{g}_+$ being the variable $\mathbf{g}$ on the right and on the left side of the interface between two elements and with $\mathbf{n}_+$ and $\mathbf{n}_-$ being the respective normal. In this case, the interface fluxes of the auxiliary variable $\mathbf{u}_{aux}^{\delta I}$ and of the primitive system $\tilde{\mathcal{H}}^{\delta I}$ are responsible for the boundary conditions to be correctly transferred into the domain” [10].

2.6 Riemann Solvers

As seen in section 2.4, the one-dimensional Riemann solvers allow the various elements of the DG/FR approach applied to the advection term of the compressible Navier-Stokes to communicate and correctly transfer the information. To propagate the information across elements we then need to solve a Riemann problem at each interface.

When solving a Riemann problem at a given interface, the two-dimensional problem is rotated into a one-dimensional problem in the normal direction with respect to the given interface. Firstly, the rotation matrix $\mathbf{R}$ is applied to the variables $\mathbf{u}$. Secondly, a Riemann solver is used to calculate the augmented one-dimensional flux ($\mathbf{f}(\mathbf{u}_n) = \mathbf{f}(\mathbf{R}\mathbf{u})$). Finally, the augmented one-dimensional flux is rotated back to the Cartesian frame of reference.

In *Nektar++* the solution to the Riemann problem can be obtained through an exact or nine approximated Riemann Solvers. The exact solver identifies the wave pattern among the well-known ten possible cases and correspondingly calculates the associated numerical flux between elements in a Godunov-type approach [11]. The rest of approximated solvers, simplify the possible wave pattern configurations. In this work, the HLLC Riemann Solver is considered. The accuracy of the exact Riemann solver enhances computational cost. The accuracy obtained for previous case studies for the HLLC was the same as the Exact solver while providing lower computational cost [12].

2.7 Implementation of the Boundary Conditions

The nature and implementation of the boundary conditions (BCs) can affect the stability and accuracy of a Computational Fluid Dynamics solver, particularly when working with compressible flows. In *Nektar++* the Compressible Flow Solver focuses on a weak treatment of the boundary conditions applied to the compressible Navier-Stokes equations, in which the BCs are imposed by modifying the state from which the numerical flux is calculated [10]. Previous publications state that weak BCs increase convergence and accuracy over strong BCs [13] [14].

Farfield Conditions, Inlet case: In the advection term of the compressible Navier-Stokes equations, the *Weak-Riemann* approach is used for applying the BCs. This method specifies all the variables at the boundary from free-stream conditions. The external (ghost) state is used as the right state. Once the left (inner) and a right state are set, it solves a Riemann problem and calculate the boundary intercell numerical flux, taking automatically into account the eigenvalues of the Euler equations [12]. This method is proved to be identical to a *Weak-Prescribed Riemann-invariant* approach. This is based on a characteristic approach, where the Riemann invariants are calculated and correspondingly applied as BCs. The solver automatically considers if the boundary is an inflow or outflow. The method is also known as no-reflective BCs as it damps the spurious reflections from the boundaries [9].

Farfield Conditions, Outlet case: The characteristic approach presented for the Euler equations for farfield boundaries, works also for the advective flux of the Navier-Stokes equations in regions where viscosity effects can be neglected, as is it the case of the inlet. However, in the outlet shedding is present, so viscosity effects become important. In this case, the characteristic treatment of the BCs generates spurious oscillations polluting the overall solution and leading to numerical instabilities. To deal with this, *Nektar++* implements a method based on the so-called sponge terms, modifying the RHS of the compressible NS equations as follows:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_1}{\partial x_1} + \frac{\partial \mathbf{f}_2}{\partial x_2} = \sigma(\bar{\mathbf{x}})(\mathbf{u}_{ref} - \mathbf{u}), \quad (2.19)$$

where $\sigma(\bar{\mathbf{x}})$ is a damping coefficient defined in a region $\bar{\mathbf{x}}$ in proximity to the boundaries and $\mathbf{u}_{ref}$ is a known reference solution. The length and the shape of the damping coefficient depend on the problem being solved [9].

In the outlet of this thesis study case viscous effects are important, which means that BCs are applied differently to recover the physical solution. The advection term is treated in a Weak-Riemann sense. *Nektar++* automatically detects the outflow is subsonic, for which the pressure

is estimated on the ghost state and the density and velocity are extrapolated from the interior of the domain. Notice, this is a reflective implementation that combined with the sponge region become fully non-reflective. Regarding the diffusion term of the Navier-Stokes equations a LDG operator is used. Therefore, the values of the auxiliary variables are directly imposed, which means the BCs are applied in a Weak-Prescribed sense [10].

2.8 De-Aliasing Strategies

Aliasing effects, arising as a consequence of the nonlinearity of the underlying problem, need to be address to stabilise the simulations. Aliasing appears when nonlinear quantities are calculated at an insufficient number of quadrature points. We can identify two types of nonlinearities:

- PDE nonlinearities, related to the nonlinear and quasi-linear fluxes.
- Geometrical nonlinearities, related to the deformed/curves meshes.

We consider two de-aliasing strategies based on the concept of consistent integration:

- Local dealiasing: It only targets the PDE-aliasing sources, applying a consistent integration of them locally.
- Global dealiasing: It targets both the PDE and the geometrical-aliasing sources. It requires a richer quadrature order to consistently integrate the nonlinear fluxes, the geometric factors, the mass matrix and the boundary term.

In this thesis, a global dealiasing technique is used. In the way in which Nektar++ is implemented, since the PDE and geometric aliasing are tackled separately during the projection and solution of the equations, to consistently integrate all the nonlinearities the quadrature points should be selected based on the maximum order of the nonlinearities:

$$Q_{min} = P_{exp} + \frac{\max(2P_{exp}, P_{geom})}{2} + \frac{3}{2}, \quad (2.20)$$

where Q_{min} is the minimum required number of quadrature points to exactly integrate the highest-degree of nonlinearity, P_{exp} being the order of the polynomial expansion and P_{geom} being the geometric order of the mesh. Bear in mind that we are using a discontinuous discretisation, meaning that aliasing effect are not fully controlled, since the boundary terms introduce non-polynomial functions into the problem [15].

2.9 Non-Smooth Artificial Viscosity

When dealing with compressible flows it is usual the presence of abrupt changes in density, often referred to as shocks. According to the Gibbs phenomena, discontinuities within the flow domain lead to numerical instabilities. To stabilise the flow, *Nektar++* utilises a shock capturing technique which makes use of artificial viscosity to damp the numerical fluctuations, in conjunction with a sensor algorithm to decide where the artificial viscosity is required. The sensor calculated the smoothness of the solution in the following way:

$$S_e = \frac{\|\rho_e^p - \rho_e^{p-1}\|_{L_2}}{\|\rho_e^p\|_{L_2}}, \quad (2.21)$$

Depending on the local value of the sensor, an artificial diffusion term is locally added to the Navier-Stokes equations.

In this work, we have made use of the non-smooth artificial viscosity model, in which the viscosity is constant in each element and discontinuous across elements. The artificial viscosity is modelled by adding a Laplacian term on the right-hand side of equation 2.1. The diffusivity is then controlled with the value of the coefficient ϵ defined as:

$$\epsilon = \epsilon_0 \begin{cases} 0 & \text{if } S_e < s_\kappa - \kappa \\ 0.5(1 + \sin \frac{\pi(S_e - s_\kappa)}{2\kappa}) & \text{if } s_\kappa - \kappa < S_e < s_\kappa + \kappa \\ 1 & \text{if } S_e > s_\kappa + \kappa \end{cases} \quad (2.22)$$

where ϵ_0 is the maximum values for the viscosity coefficient, κ is half of the width of the transition interval and s_κ is the value of the centre of the interval [16].

Chapter 3

Problem Formulation

In this chapter, the problem formulation is described. Firstly, the flow geometry and mesh used to perform Direct Numerical Simulation are presented. Secondly, the boundary and the initial conditions are outlined. Finally, the solver configuration is described.

3.1 Flow Geometry and Mesh

The T106A low-pressure turbine cascade has been studied extensively both in experiments [17] [18], and numerical simulations [19] [2] [20]. The T106A profile represents the mid-span of the Pratt and Whitney PW2037 rotor airfoil [21]. The geometrical details are shown in Table 3.1 and the blade geometry is illustrated in Figure 3.1 [22]:

Chord	198mm
Axial chord	170mm
Blade stagger	30.7°
Pitch	158 mm
Span	375 mm
Suction surface length	264.7 mm
Pressure surface length	230 mm
Design exit flow angle	63.2°

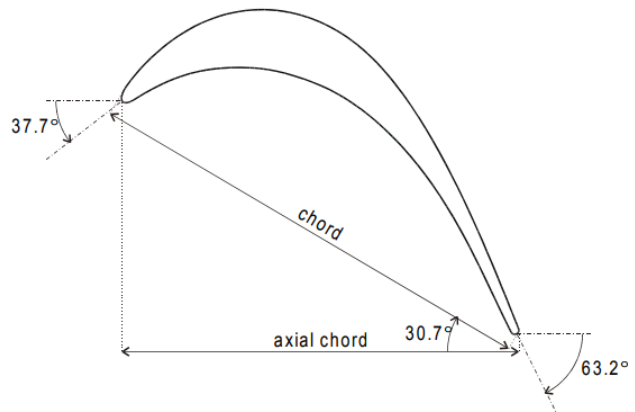


Table 3.1 and Figure 3.1: T106A low-pressure blade cascade specifications from Stieger [22].

The mid-span section of a single passage of the cascade is meshed, using periodic boundary conditions both in the pitch and span directions. It is important to highlight the use of a mixed mesh, consisting of triangular and quadrilateral elements.

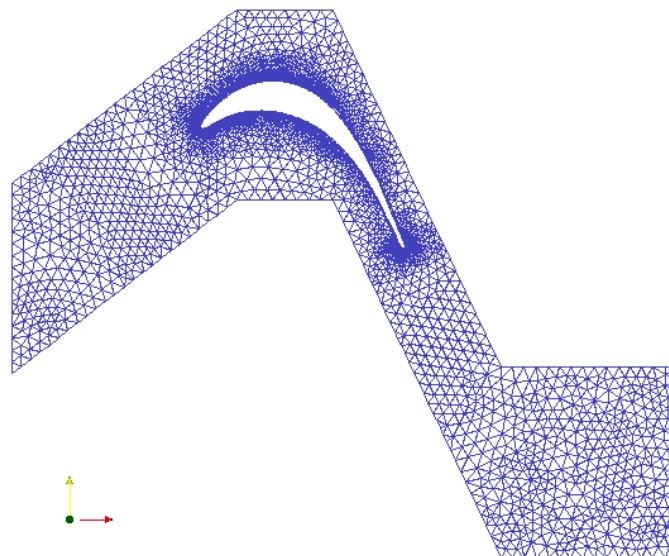


Figure 3.2: Complete view of the computational domain and mesh.

The resolution requirements in near-wall regions is an important matter. The boundary layer is captured when the cell in the boundary layer is of the order of the dimensionless wall distance y^+ . To facilitate this, the mesh is refined in the surface of the turbine blade as described in Table 3.2. Note the approximated value of wall spacing of the mesh Δy , used in section 3.1.1, is 0.002.

Parameters	Value
Δs_{min}	0.02
Δs_{max}	0.05
Order	6
Boundary Layer Thickness	0.008
Boundary Layer Layers	4
Boundary Layer Progression	1.1

Table 3.2: Mesh conditions.

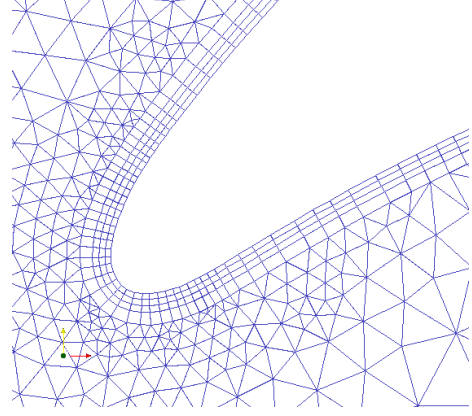


Figure 3.3: Boundary layer of the mesh.

3.1.1 Spatial Resolution for Direct Numerical Simulation

When aiming direct numerical simulation, spatial resolution is a crucial factor. The smallest length scale is captured when the mesh spacing in wall units meets the following criteria [23]:

$$\Delta x_{wall}^+ = \frac{\Delta x}{\delta_v P} < 20 \quad , \quad \Delta y_{wall}^+ = \frac{\Delta y}{\delta_v P} < 1 \quad , \quad (3.1)$$

where P is the polynomial order of the expansion basis and Δx and Δy are the wall spacing of the mesh in the tangent and normal directions to the blade surface respectively. The viscous length scale δ_v is calculated at each point of the blade by extracting the local shear stress and density along the blade:

$$\delta_v \equiv \nu \sqrt{\frac{\rho}{\tau_w}} \quad (3.2)$$

In this work, a low Reynold number of 8,000 is firstly imposed and then it is increased to a moderate Reynolds number of 60,000. The Kolmogorov length scale is inversely proportional to the Reynolds number. However, the mesh used for both cases is identical. Thus, an increment of the polynomial order for the higher Reynolds number is required to resolve all the spatial length scales. This fact illustrates the benefit of using the spectral/hp element method, less number of elements are required to achieve the same level of resolution than in low-order methods due to an enhanced resolution power. The polynomial order for each case is tuned to provide adequate spatial resolution. The polynomial expansion order P used for the Reynolds number of 8,000 cases is 6. For the Reynolds number of 60,000 and Mach 0.1 the polynomial order is 13.

The spatial resolution for the Reynolds 8,000 cases is illustrated in Figure 3.4 and 3.5. Figure 3.6 displays the spatial resolution at Reynolds 60,000. The difference in the smoothness in both directions is due to a constant wall spacing Δy and a varying Δx .

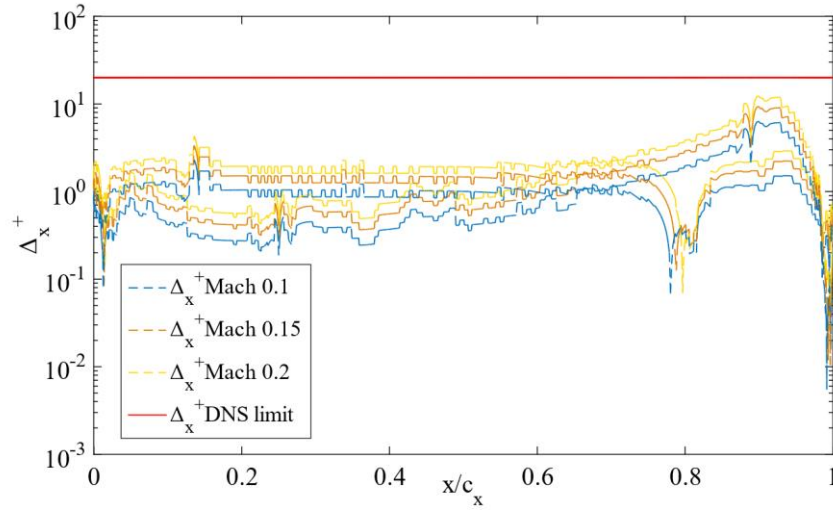


Figure 3.4: DNS Spatial Resolution in x for Reynolds number of 8,000.

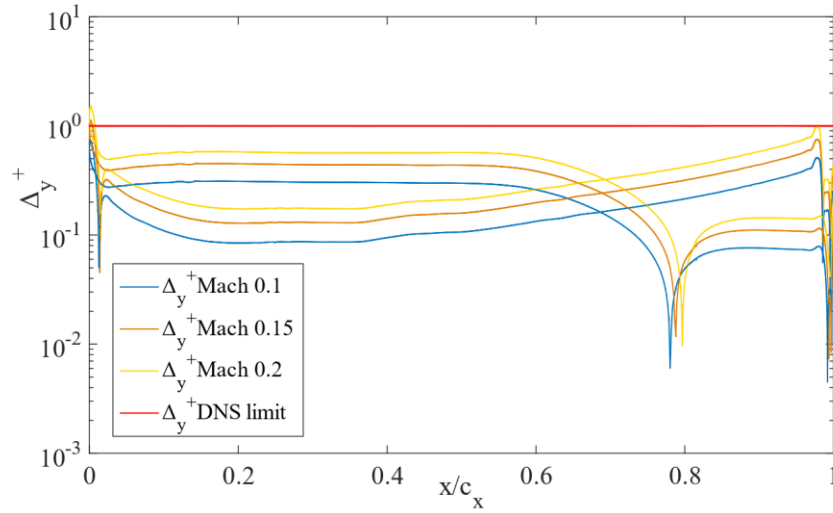


Figure 3.5: DNS Spatial Resolution in y for Reynolds number of 8,000.

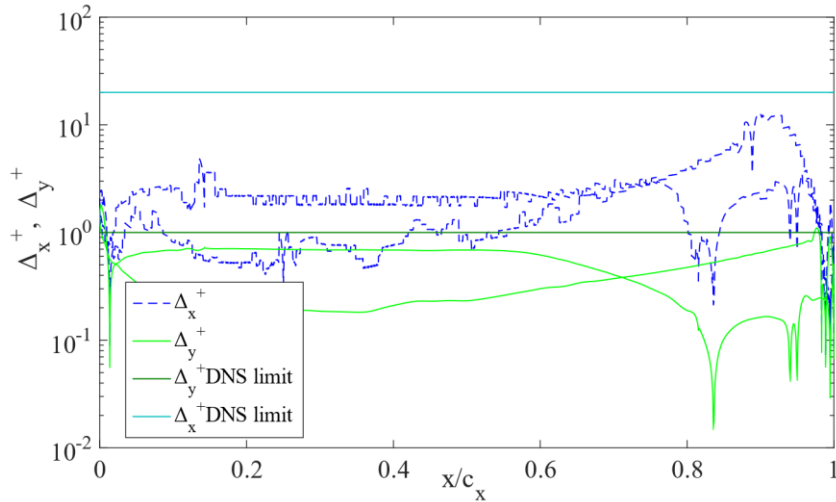


Figure 3.6: DNS Spatial Resolution in x and y for Reynolds number of 60,000.

3.1.2 Quadrature Points

When studying relevant industry problems, complex geometries comprised of curved elements are present. This gives rise to global aliasing sources as explained in section 2.8. The global de-aliasing technique is used to tackle both the PDE and geometric aliasing. As defined in the Table 3.2, the geometric mesh order P_{geom} is equal to 6. Consequently, the minimum number of quadrature points to consistently integrate the highest-degree of nonlinearity is:

$$Q_{min} = P_{exp} + \frac{\max(2P_{exp}, 6)}{2} + \frac{3}{2}, \quad (3.3)$$

P_{exp} corresponds to the order of the polynomial expansion, this is 6 in the case of a Reynolds number of 8,000 and 13 in the case of 60,000. Consequently, the number of quadrature points used are $Q = 14$ at a Reynolds 8,000 and $Q = 28$ at a Reynolds of 60,000.

Even though global aliasing effects were tackled, the fact that we are using a discontinuous discretisation means that aliasing effects are not fully under control, since the boundary terms introduce non-polynomial functions into the problem. This means that, although de-aliasing techniques are enough to stabilise the flow in multiple research studied, in the present study stability is not guaranteed [15].

3.2 Boundary Conditions

The real flow in turbine blades of a gas turbine is unsteady with convected disturbances from upstream components and potential disturbances from upstream and downstream. The rotor-stator interaction arises from the convection of wakes shed from upstream blade rows. This periodic shedding causes unsteady effects on the blade surface boundary layers and has been proved to have important effects downstream, inspiring numerous studies [21]. However, in this thesis, the upstream wakes are neglected, where a clean steady inlet condition is defined. Therefore, this is only a good approximation for the first blade row of the turbine.

In this work, various case studies have been compared. At the inlet, two different Reynolds numbers have been studied, a Reynolds number of 8,000 and 60,000. The inlet Mach number cases to assess the compressibility effects are 0.1, 0.15 and 0.2. The Prandtl number selected is 0.72. Finally, the inlet flow angle is chosen as 37.7° in all the cases to be consistent with previous experiments [22] [24]. In the Compressible Flow Solver, the boundary conditions are weakly implemented, which means that the BCs are applied to the fluxes. As a result, the pressure and the density in the inlet are required. The values for the inlet pressure and density are chosen according to the T106C turbine blade cascade previously run in *Nektar++* (MMC S11 case [8]).

For the compressible Navier-Stokes equations a non-slip condition must be applied to the velocity field at a solid wall, which corresponds to the blade for this problem. The blade wall is defined as an isothermal wall with imposed temperature $T_{wall} = 146.21K$.

Finally, note that periodic boundary condition for all flow quantities is imposed in the pitch and span directions to represent a turbine cascade configuration.

3.3 Initial Conditions

When starting the Compressible Flow solver with an initial condition that differs from the solution, waves initially develop giving stability problems (see Figure 3.7). The numerical instabilities caused by the flow discontinuities were controlled by introducing artificial viscosity. Throughout the numerous cases run in this thesis, the smooth artificial viscosity model failed to produce stable simulations. Otherwise, the non-smooth model managed to alleviate stability issues. The parameters controlling the diffusivity coefficients were set to 1 for the maximum value for the viscosity coefficient ϵ_0 ; -1.3 for the value of the centre of the interval s_k and 0.2 corresponding to half the width of the transition interval.

In the Reynolds number of 8,000 case, the flow is impulsively started as no previous simulations were performed. For Reynolds number case of 60,000 the flow obtained from the 8,000 Reynolds number is set as initial conditions.

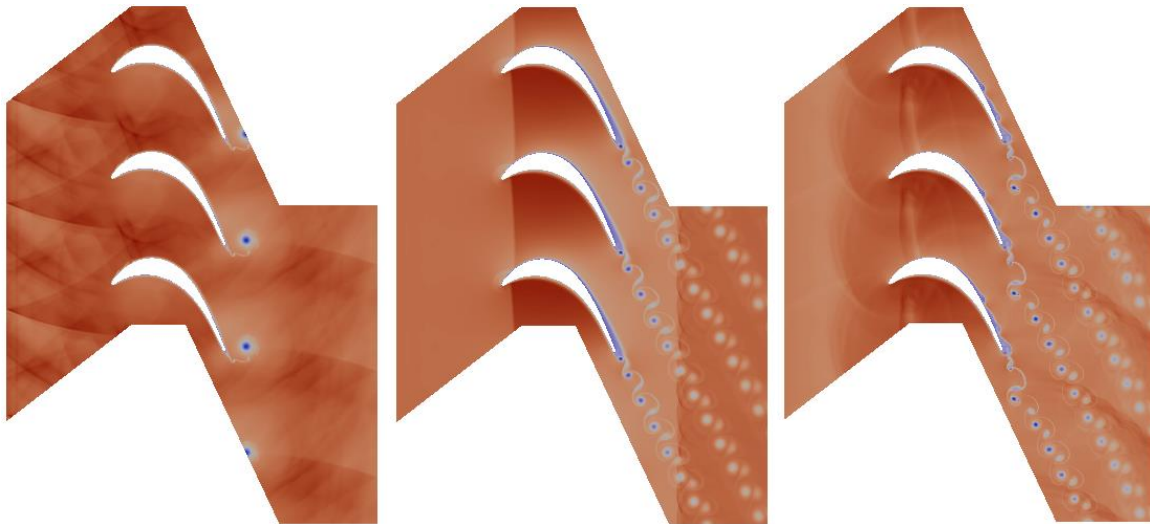


Figure 3.7: Instantaneous density waves created when initializing the flow. a) Reynolds number of 8,000 b) and c) Reynolds of 60,000.

The simulations were run until the flow stabilises and the discontinuities disappear. This section assesses the time required to run the simulation before the stabilised output flow solution is set as initial conditions. Apart from visualizing the instantaneous flow field, the kinetic energy trend was plotted against the simulation time. In this thesis, the simulation time is expressed in terms

of equivalent chord lengths. The kinetic energy graph helped to determine the number of equivalent chord lengths required for stabilising the flow and avoiding the average quantities to be polluted. The kinetic energy evolution and the visualization of the instantaneous flow field suggested that the flow can be re-initialized after four equivalent chord lengths.

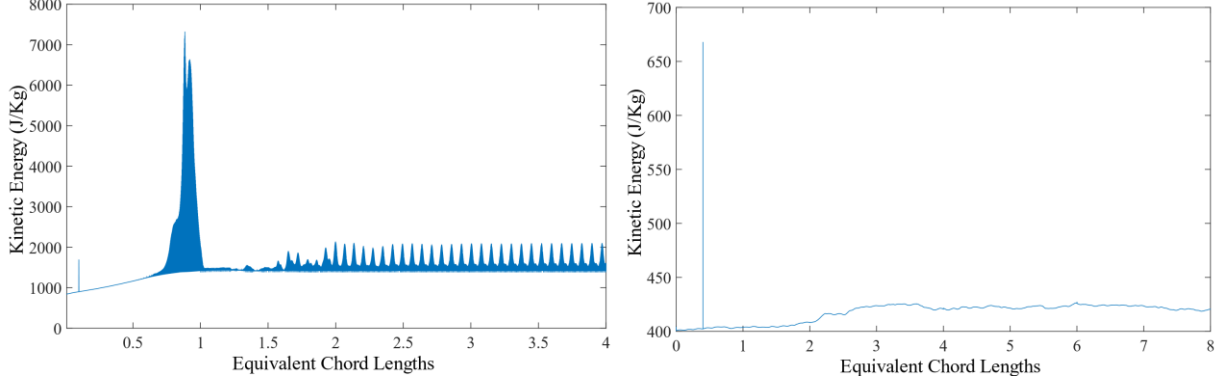


Figure 3.8: Kinetic Energy Evolution. a) Reynolds number 8,000. B) Reynolds number 60,000.

3.4 Solver Configuration

This section declares how the flow will be solved by specifying the equations to solve, the projection type, the advection and the diffusion operators used, the Riemann solver implemented, the time-integration scheme, and the parameters of the artificial viscosity. In addition, the time average filter setting is introduced.

3.4.1 Equations and Projection Type

The effects of fluid viscosity and heat conduction as well as compressible effects are included. Consequently, the equations to solve are the compressible Navier-Stokes equations. In *Nektar++* the spatial discretisation of the compressible Navier-Stokes equations is projected in the polynomial space via discontinuous projection. Specifically, we make use either of the discontinuous Galerkin or the Flux-Reconstruction approach. Consequently, the Projection type is set to discontinuous as continuous projection is not supported in the Compressible Flow Solver.

3.4.2 Advection and Diffusion Operators

In order to reduce the computational cost, it was attempted to use the FR_{G2} scheme. This scheme would allow for a twice bigger time step and a consistent speed-up of the simulations performed. The coupling of these schemes with the dealiasing techniques would permit maintaining numerical stability, thus preventing the simulations to fail due to aliasing-drive instabilities. However, FR_{G2} is not implemented for triangular elements. The use of a mixed elements mesh lead to the selection of the classical Discontinuous Galerkin in weak form. As per section 3.2, a global de-aliasing technique is implemented, where the number of quadrature points Q is larger

than the polynomial order inside each element P . This scheme is denoted by $DG_{Q>P} - EMM$. The term exact mass matrix (EMM) refers to when solving a fully linear problem [12]. This method is the most accurate but also the most restrictive in terms of time-step. In this thesis, the diffusion term was discretised with the Local Discontinuous Galerkin approach (LDG) presented in section 2.5.

3.4.3 Time-integration Scheme

A critical aspect of the Compressible solver is that it only supports explicit time-integration schemes. The stability of such schemes is governed by the CFL condition, which requires the space-time analytical domain of dependence to be contained within the numerical one.

The evaluation of the time-step limit restriction consists in bounding the largest eigenvalue λ_{max} of the advection operator [9].

$$\Delta t \cdot \lambda_{max} \leq const \quad (3.4)$$

The maximum eigenvalue in a general domain follows:

$$C_\lambda(\mathbf{v}, h)P^2 > \lambda_{max} \quad (3.5)$$

being h the characteristic length, $\mathbf{v}$ the local advection velocity and P is the polynomial order of the solution. The time-step restriction is bounded then by:

$$\Delta t \leq \frac{\alpha}{C_\lambda(\mathbf{v}, h)P^2} \quad (3.6)$$

where α represents the distance from the origin of the boundary of the stability region along the azimuthal of the dominating eigenvalue λ_{max} , and $C_\lambda(\mathbf{v}, h)$ can be estimated by:

$$C_\lambda(\mathbf{v}, h) \approx \frac{c_\lambda}{h} \max|\mathbf{v}^e| \quad (3.7)$$

where c_λ is a constant [7], $\mathbf{v}^e$ corresponds to the velocity in the element e and h is the size of the element. Combining equations 3.6 and 3.7 the time step restriction follows:

$$\max|\mathbf{v}^e| \frac{c_\lambda}{h} P^2 \Delta t \leq \alpha \quad (3.8)$$

Thus, the time step restriction is directly proportional to the mesh spacing and inversely proportional to the square of the polynomial order used, which makes it a very restrictive requirement in the case of high Reynolds numbers.

For the time-step to ensure that the error from the time-stepping scheme is negligible compared to the spatial discretisation, it needs to be 100 times lower than the time-step limit imposed by the Courant-Friedrichs-Lewy condition [25]. This is however, a very demanding time-

step restriction, only feasible for short final output times, as cases studying the connections between different schemes [12] [9]. In this work, the initial conditions are set after 4 equivalent chord lengths, and the flow field variables are time averaged over 4 equivalent chord lengths. The longer simulation time makes unfeasible the use of such restrictive time step. Hence, the time-step was selected to be three times smaller than the maximum time step allowing the simulation to remain stable (Δt_{max}) [12].

For all the simulations, the time integration method implemented is the 4th order Runge-Kutta. Even though this method is more expensive, the 4th order Runge-Kutta was selected because of its more satisfactory stability region for many benchmark studies [12][9].

3.4.4 Riemann Solver for the Advection Operator

As described in section 2.6, a Riemann problem is solved at each interface of the computational domain for the advection term. *Nektar + +* provides ten different Riemann solvers, one exact and nine approximated. The Exact Toro solver, solves the conservation problem by using a Newton iterative method. The order of accuracy of the exact solver is related to the residual in the Newton method, which implies higher computational cost. On the other hand, the approximated Riemann solvers do not take into account the full Riemann problem, these simplifications of the exact solver provide lower computational cost but lower accuracy [16]. In this thesis, the numerical flux has been calculated through the HLLC (Harten, Lax, van Leer + Contact) Riemann solver. The reasons for the choice being the lower computational time against the Exact Toro while maintaining a similar accuracy. The accuracy of the Exact Toro and the HLLC Riemann solvers have been proved the same in several benchmark tests [9].

3.4.6 Time Averaging Filter

After initial conditions are set following results in section 3.3, one needs to assess the time required to dump averages. This section explains the process to select the time to average the unsteady flow quantities.

The average flow field was computed over four consecutive equivalent chord lengths. Apart from the averaged flow field accumulated over the four equivalent chord lengths, two more averaged flow fields were generated. One averaged flow field was created after the first two equivalent chord lengths and a second average flow field of the last two chord lengths. The averaged quantities of the three averaged fields were compared ensuring convergence when averaging over four equivalent chord lengths.

The difference between the pressure coefficient averaged over two and four equivalent chord lengths was analysed in Figure 3.9. The maximum error when averaging over two equivalent chord lengths is 2.2% the value of the maximum pressure coefficient. Thus, the error averaging over four equivalent chord lengths is lower.

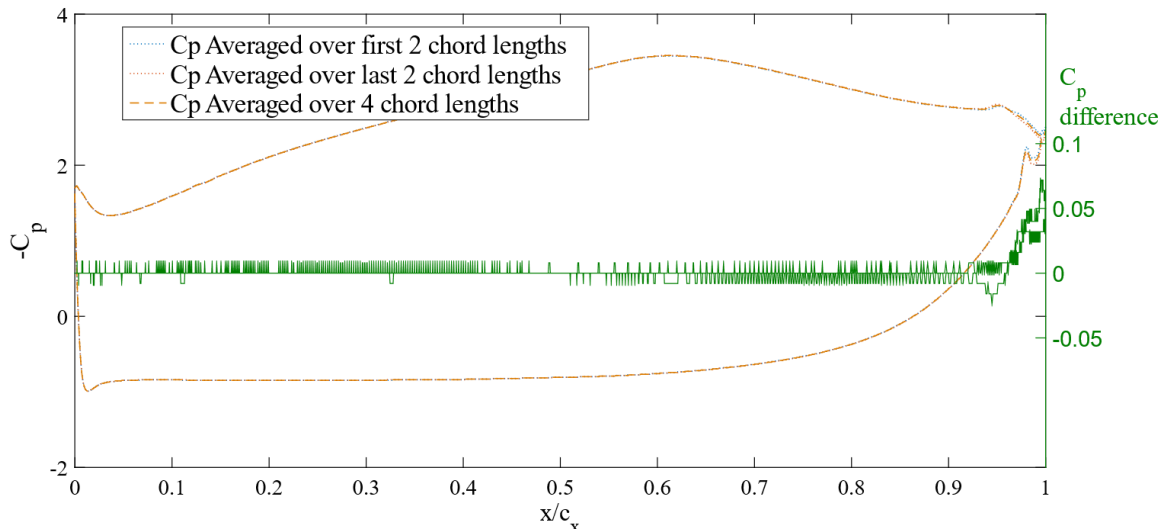


Figure 3.9: Time averaged pressure coefficient with varying averaging time at Reynolds 60,000.

For the friction coefficient, the maximum difference is less than 1% the value of the maximum friction coefficient. Results show that four equivalent chord lengths are enough to achieve convergence of the averaged flow fields at a Reynolds number of 60,000. Also, results confirm that the initial conditions are well selected as they are not affecting the time average results.

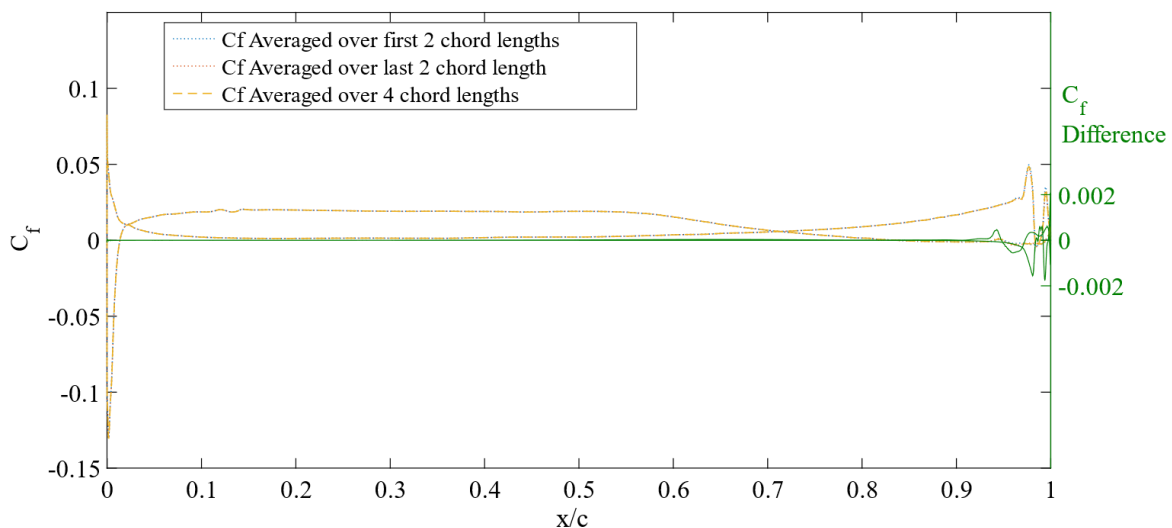


Figure 3.10: Time averaged friction coefficient with varying averaging time at Reynolds 60,000.

Chapter 4

Simulation Results

This thesis assesses the importance of moving towards a more accurate result by running the simulations over the T106A blade cascade with the Compressible Solver instead of with the Incompressible Navier Stokes Solver. Firstly, the importance of compressibility effects is analysed by comparing the Incompressible and Compressible Solver results under three different Mach numbers of 0.1, 0.15 and 0.2. The simulations are performed under a low Reynolds number of 8,000 and subsequently, the Reynolds number is increased to 60,000. This thesis evaluates the time-averaged pressure and friction coefficients, the time-averaged boundary layer profile and the boundary layer parameters along the blade. The study of these parameters concludes that compressibility effects play an important role. However, the use of the Compressible Flow Solver presents three main downsides. It requires larger computational time, the inlet specifications are less precise and the user-guide is incomplete.

4.4 Compressibility Effects

Initially, a low Reynolds number of 8,000 is used under different Mach numbers to understand the importance of the compressibility effects. The results obtained with the Incompressible and Compressible solvers are compared under three different Mach numbers of 0.1, 0.15 and 0.2. The comparison shows how the solution given by the Compressible Flow solver converges to the solution of the Incompressible solver as the compressibility effects become negligible. It is concluded that compressibility effects play an important role in the study of the T106A turbine blade cascade even when the Mach number is low. Firstly, the pressure and friction coefficients are compared. Secondly, the separation point difference by both solvers is examined. Finally, the boundary layer profiles and parameters along the blade are studied.

4.4.1 Pressure Coefficient

The pressure coefficient ($C_p = \frac{p - p_\infty}{\frac{1}{2} \rho_\infty U_\infty^2}$) obtained by the Incompressible and Compressible solvers for the three different Mach numbers is presented in Figure 4.1 and Figure 4.2. Note that, the pressure coefficient result obtained by both solvers diverges towards the leading edge at the suction surface. The difference between the pressure along the blade for the three different Mach numbers is shown in Figure 4.2b. The difference is expressed relatively to the maximum value of the pressure coefficient. Predictably, as the Mach number increases, the compressibility effects become more important and the solutions greatly differ. For the Mach number of 0.2 the maximum difference is more than 20% of the maximum C_p value. As the Mach number reduces,

the Compressible and Incompressible results tend to converge. However, even for the lowest Mach number of 0.1 the difference is above 5% towards the trailing edge of the suction surface.

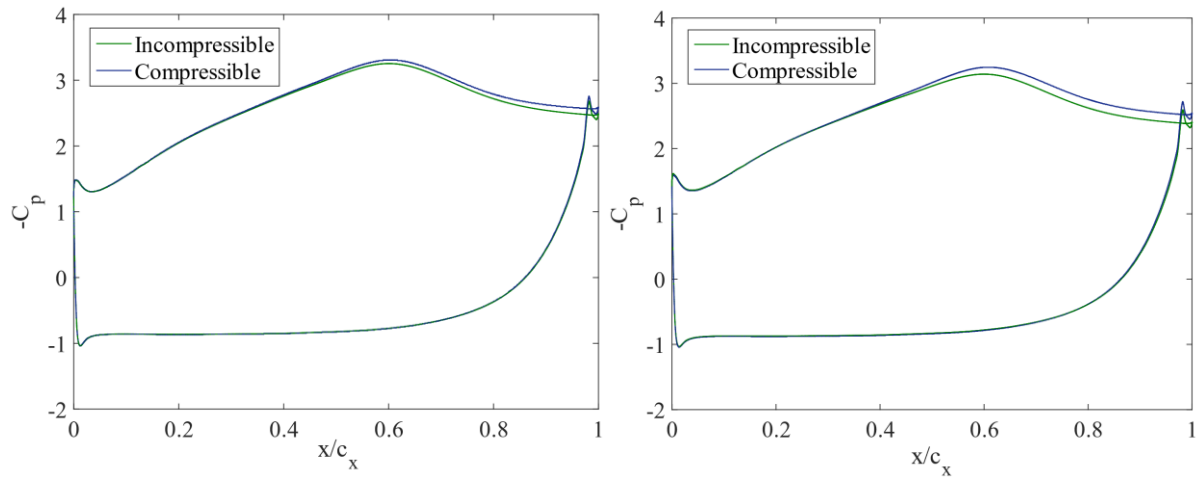


Figure 4.1: Averaged pressure coefficient along the surface of a T106A turbine blade at Reynolds 8,000

a) Mach number 0.1 b) Mach number 0.15

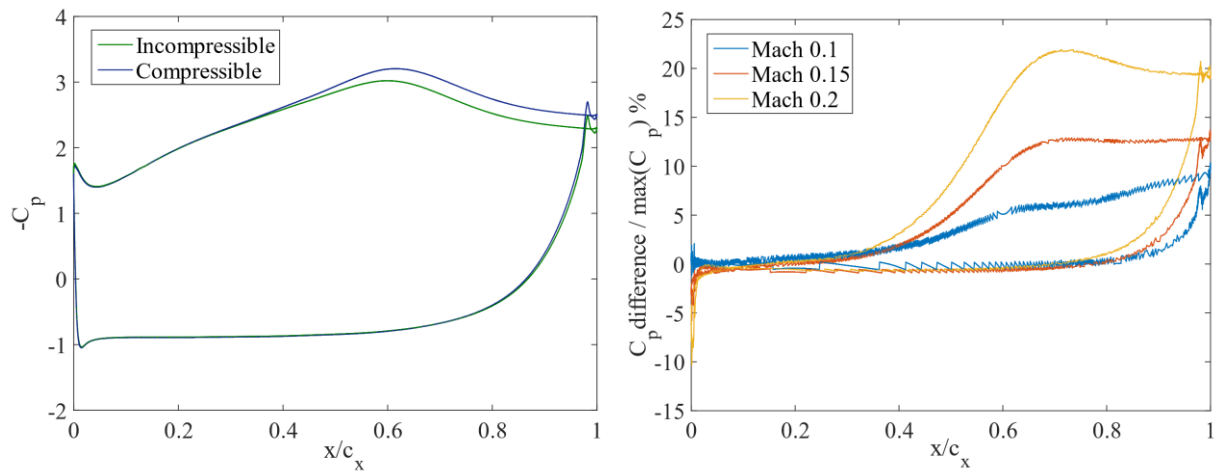


Figure 4.2: Averaged pressure coefficient along the surface of a T106A turbine blade at Reynolds 8,000

a) Mach number 0.2 b) Difference between Incompressible and Compressible solver.

4.4.2 Friction Coefficient and Separation Point

Figure 4.3 and 4.4 illustrate the time-average skin friction coefficient ($C_f = \frac{\|\tau\|}{\frac{1}{2}\rho_\infty U_\infty^2}$) obtained by both solvers for the three different Mach number cases. The friction coefficient calculated by means of both solvers present more similar results than the pressure coefficient, being almost identical for the Mach number 0.1 case and slightly varying for the Mach number 0.2 case. As a methodology to quantify this variance, the location of the separation point for the Incompressible and Compressible solver was compared. The separation point for a two-dimensional flow corresponds to the point of zero wall shear stress. Beware that, as the Reynolds number increases, specifically, for the high Reynolds number case of 60,000, the position of the

separation point becomes unstable. Throughout this thesis, the average position of the separation point is obtained by analysing the time average friction coefficient.

Figure 4.4b represents the difference in the separation position obtained by both solvers. The deviation is expressed as the ratio between the incompressible and the compressible results. The difference in the separation point location is shown to be within a 1.5% for the three cases. Predictably, the highest difference is obtained with a Mach number of 0.2. However, the lowest difference is not obtained with the lowest Mach number case but with the Mach 0.15. It is important to emphasize that, although the difference in percentage is less than in the pressure coefficient case, a variation of a 1.5% in the position of the separation point could have critical impact in the resultant drag and affect the blade re-design. Consequently, the compressibility effects are once more proved to be of utmost important with increasing Mach numbers.

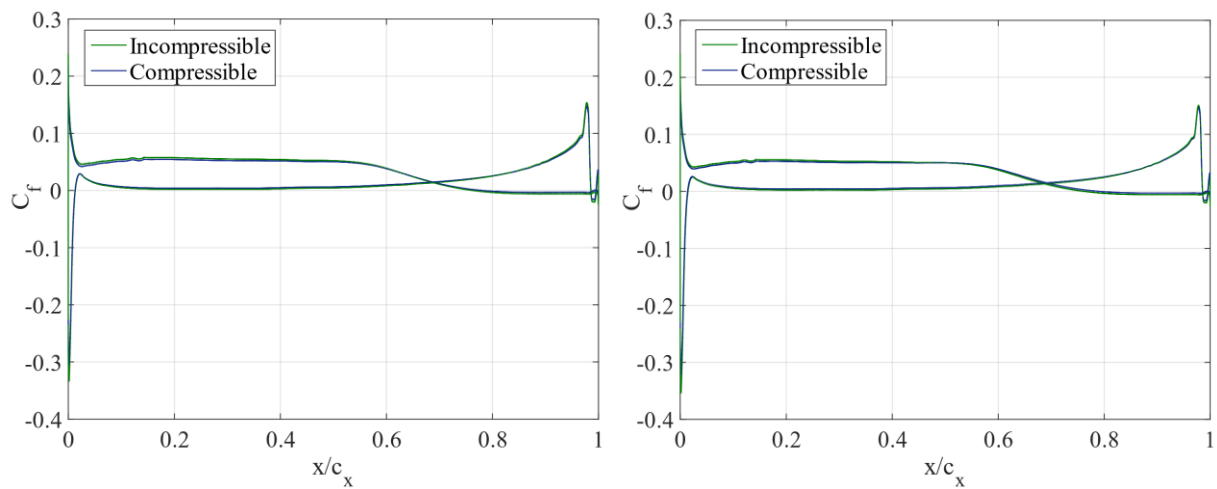


Figure 4.3: Averaged friction coefficient along the surface of a T106A turbine blade at Reynolds 8,000

a) Mach number 0.1 b) Mach number 0.15

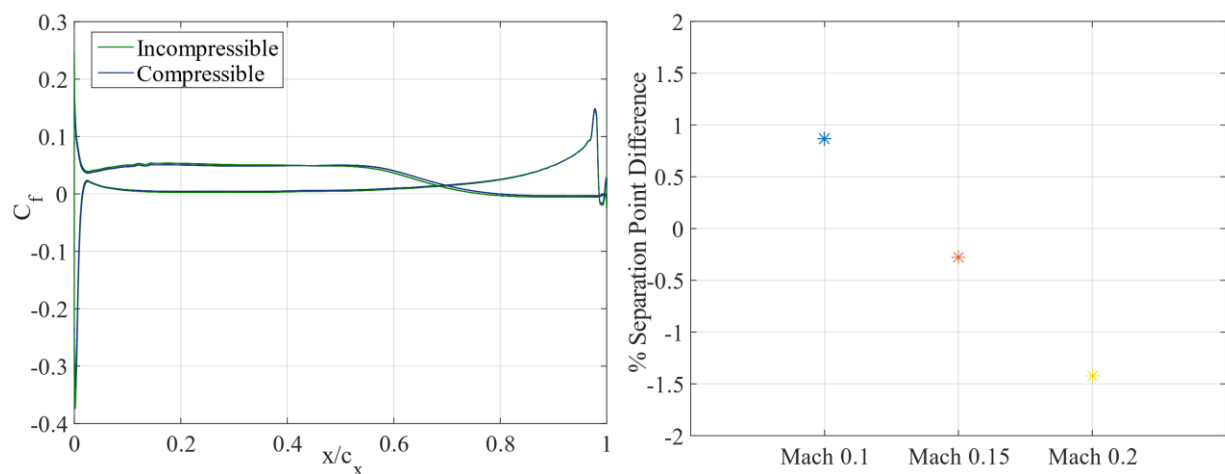


Figure 4.4: Averaged friction coefficient along the surface of a T106A turbine blade at Reynolds 8,000

a) Mach number 0.2 b) Difference of the separation point location calculated with both solvers.

4.4.3 Boundary Layer Velocity Profile

The boundary layer evolution computed by both solvers is compared along the suction surface of the blade. It is shown how for the three Mach number cases, the velocity profile greatly differs towards the trailing edge (see Figure 4.6, 4.7 and 4.8). In accordance to the previous results, the difference increases with the Mach number. In Figure 4.5a, the velocity profile is represented at $x=0$ for the three Mach numbers. At this blade coordinate, the velocity profiles for the incompressible and compressible case coincide. In Figure 4.5b, the velocity profiles are represented at $x/c_x = 0.71$. In this figure, it is noticeable the increment in the difference between both solvers results as the Mach number grows. Notice in Figure 4.5a that, the velocity at the inlet are 20, 30 and 40 m/s corresponding to a Mach number of 0.1, 0.15 and 0.2 respectively. In Figure 4.5b, appreciate that for the Mach number 0.1, the velocity increments to 40m/s, which would correspond to a Mach number of 0.2. In the case of Mach number 0.2, the velocity has reached 60 m/s which would correspond to a Mach number of 0.3. Finally, for the inlet Mach number equal to 0.2, the velocity increases to 80m/s, which corresponds to a Mach number of 0.4. As widely known, compressibility effects cannot be neglected when the Mach number is above 0.3. Therefore, the differences given by the solvers can be associated to compressibility effects. Moreover, the compressible solver presents higher velocities in the suction surface, which agrees with a lower pressure coefficient as shown in section 4.4.1. Again, compressibility effects are proved to be of extreme importance for the T106A turbine blade case.

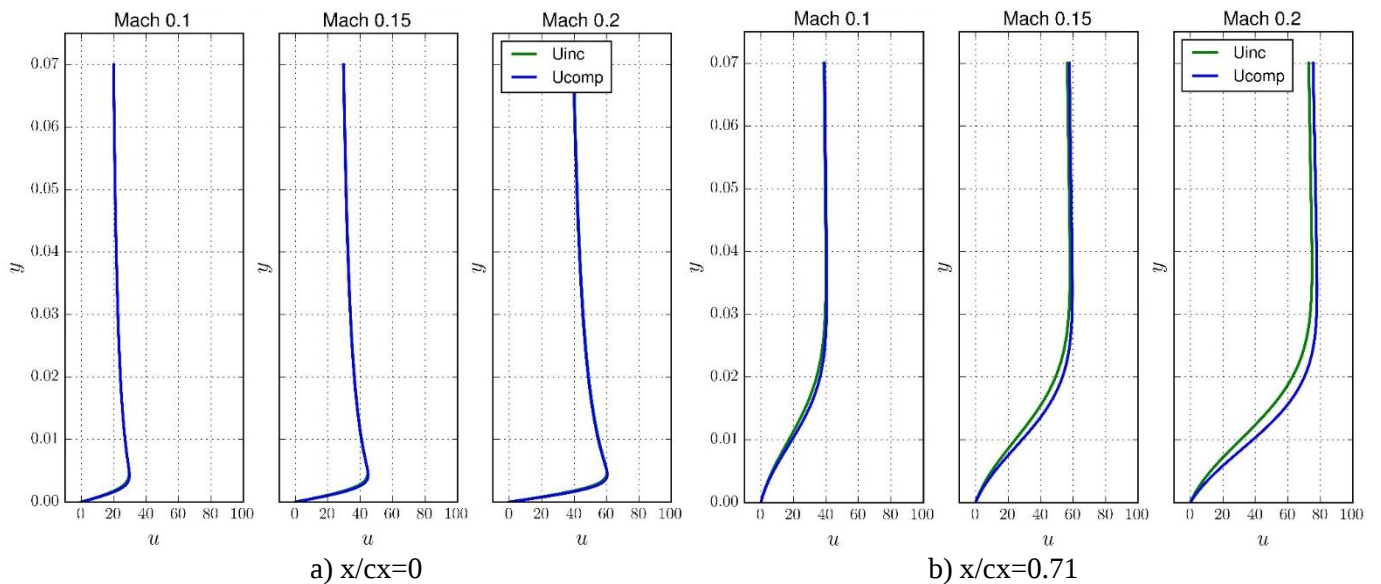


Figure 4.5: Velocity Profile Comparison for Reynolds number 8,000. a) $x/c_x=0$ b) $x/c_x=0.71$

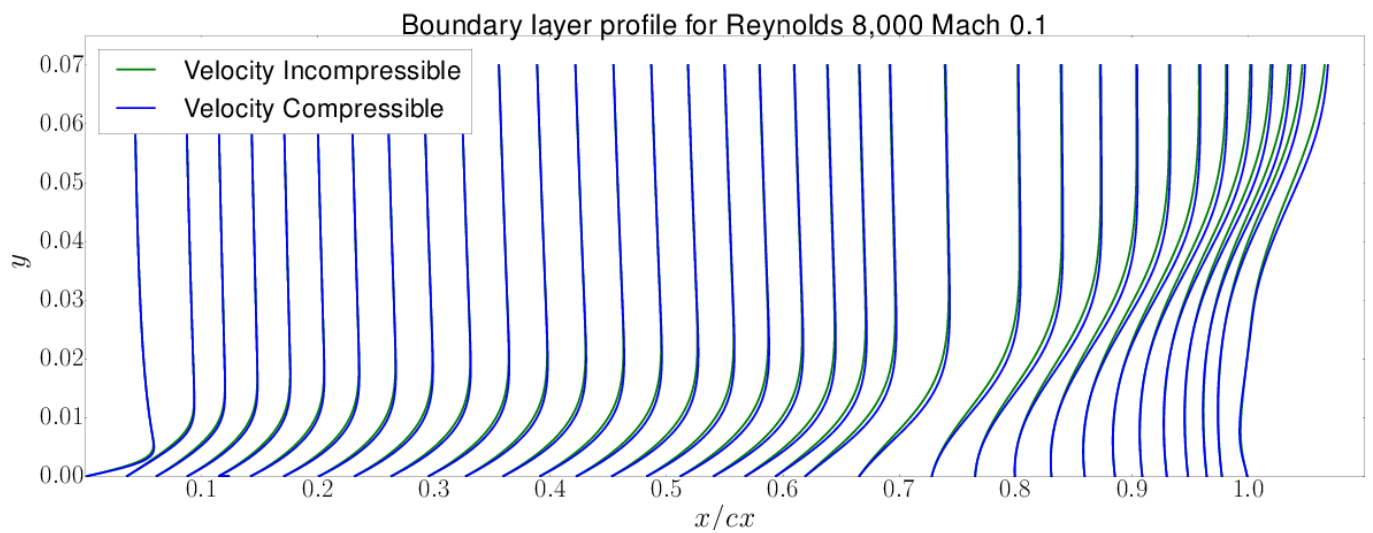


Figure 4.6: Velocity Profile Comparison for Reynolds number 8,000 and Mach 0.1.

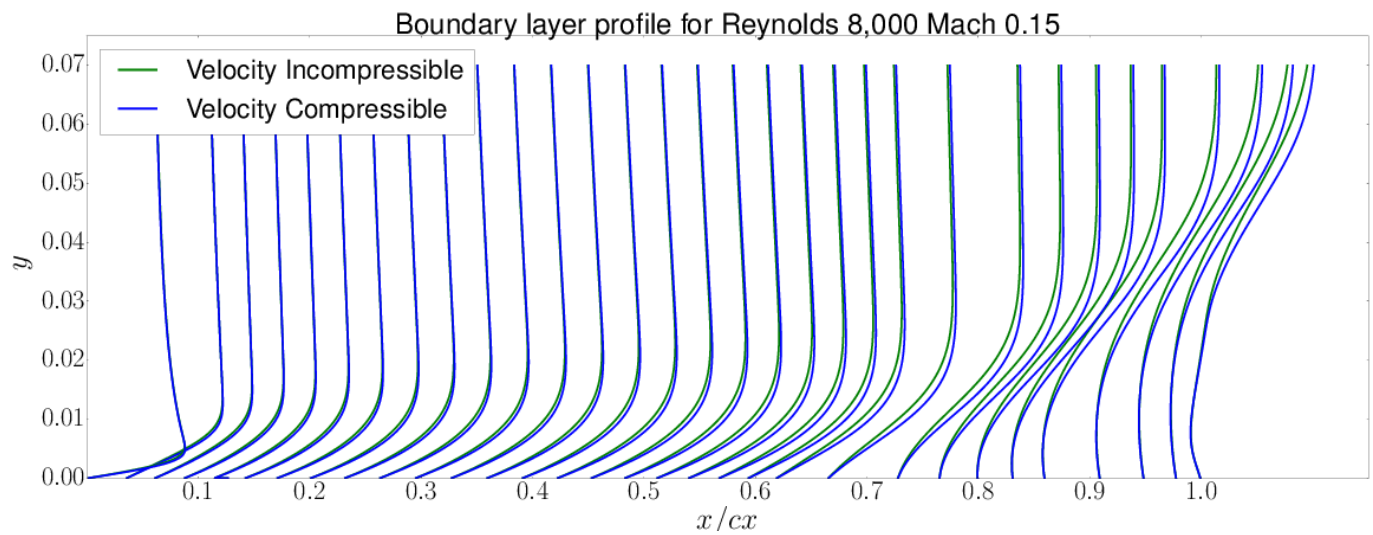


Figure 4.7: Velocity Profile Comparison for Reynolds number 8,000 and Mach 0.15.

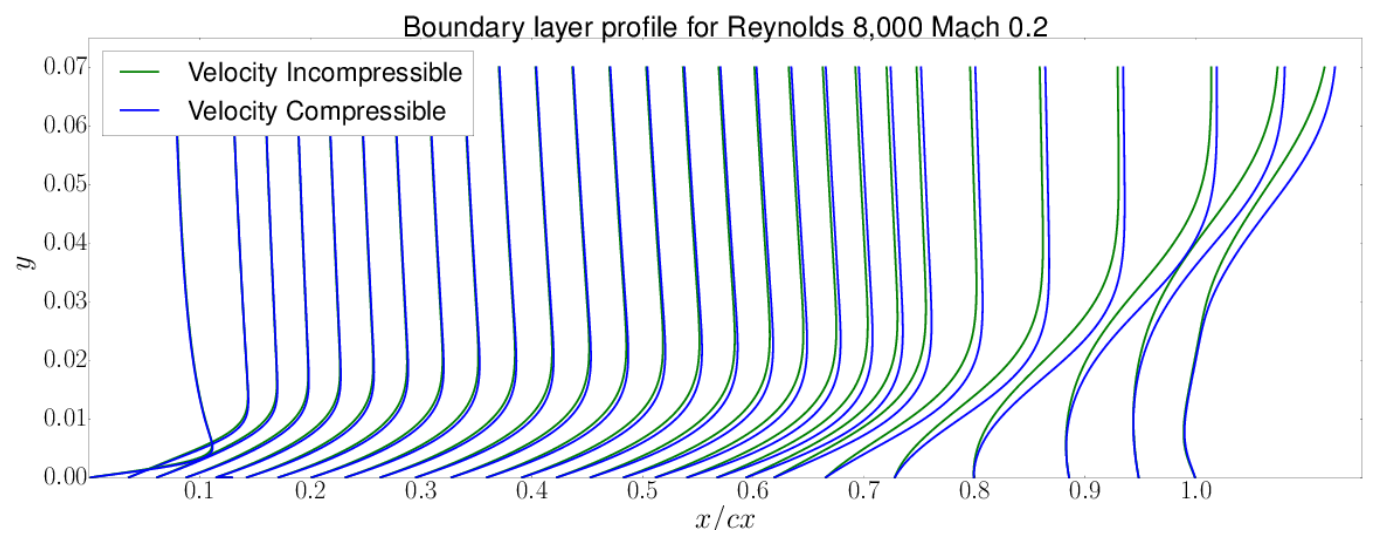


Figure 4.8: Velocity Profile Comparison for Reynolds number 8,000 and Mach 0.2.

4.4.4 Boundary Layer Parameters

The analysis of the boundary layer is based on a pseudo-velocity defined as:

$$\mathbf{u}^*(\mathbf{s}, n) := \int_0^n (\boldsymbol{\omega} \times \hat{\mathbf{n}}) dn', \quad (4.1)$$

where $\boldsymbol{\omega}$ denotes vorticity, $(\mathbf{s}, n)$ is the set of curvilinear coordinates associated to the airfoil surface. In particular, $\mathbf{s} = (s_1, s_2)$ and n are the coordinates along the streamwise, cross-flow and outward normal to the airfoil directions, respectively. The unit vector associated to these coordinates are denoted by $\hat{\mathbf{s}}_1$, $\hat{\mathbf{s}}_2$ and $\hat{\mathbf{n}}$ [26]. The benefit of using the pseudo-velocity is that as it tends to a constant outside the boundary layer, the edge of the boundary layer becomes a well-defined location [27].

The BL edge n_e is calculated as the first location satisfying:

$$\|\overline{\boldsymbol{\omega}}\| n < \epsilon \|\overline{\mathbf{u}^*}\|, \quad (4.2)$$

where the overbar denotes temporal averaging and ϵ is a tuned constant set to 0.01 for a systematic and robust detection of the BL edge [27].

The boundary layer structure is analysed in terms of the displacement thickness δ^* , the momentum thickness θ and the shape parameter H . These parameters are defined as:

$$\delta^* = \int_0^{n_e} \left(1 - \frac{\mathbf{u}_1}{u_e}\right) dn, \quad \theta = \int_0^{n_e} \left(1 - \frac{\mathbf{u}_1}{u_e}\right) \frac{\mathbf{u}_1}{u_e} dn, \quad H = \frac{\delta^*}{\theta}, \quad (4.3)$$

where $\mathbf{u}_1$ corresponds to the stream-wise velocity and u_e is the magnitude of the edge velocity.

As seen in section 4.4.3, the velocity profiles for the Incompressible and Compressible differs towards the trailing edge, and this variance grows with the Mach number. The vorticity variance should follow the same behaviour. Consequently, as the boundary layer parameters are based on the velocity and vorticity values, we expect them to match at the leading edge and progressively disagree towards the trailing edge. The contrast being augmented for higher Mach numbers when higher compressibility effects are present.

The displacement thickness and the momentum thickness results obtained for the Reynolds case of 8,000 are presented in Figure 4.9a, Figure 4.10a and Figure 4.11a. The variance observations match with the expectations presented above. Figure 4.9b, 4.10b and 4.11b represent the shape parameter for the different Mach numbers along the chord of the blade. Note that the BL separation leads to an increase in the displacement thickness and thus the shape parameter for the suction surface.

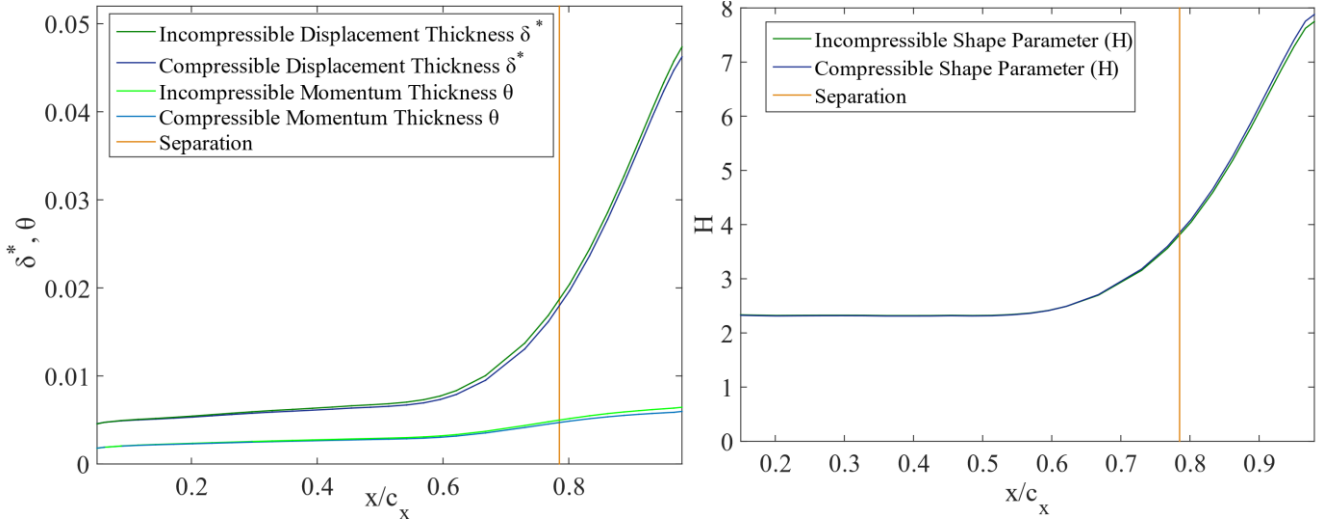


Figure 4.9: Boundary layer parameters comparison between solvers at Reynolds 8,000 and Mach 0.1.

a) Displacement and momentum thickness. b) Shape parameter.

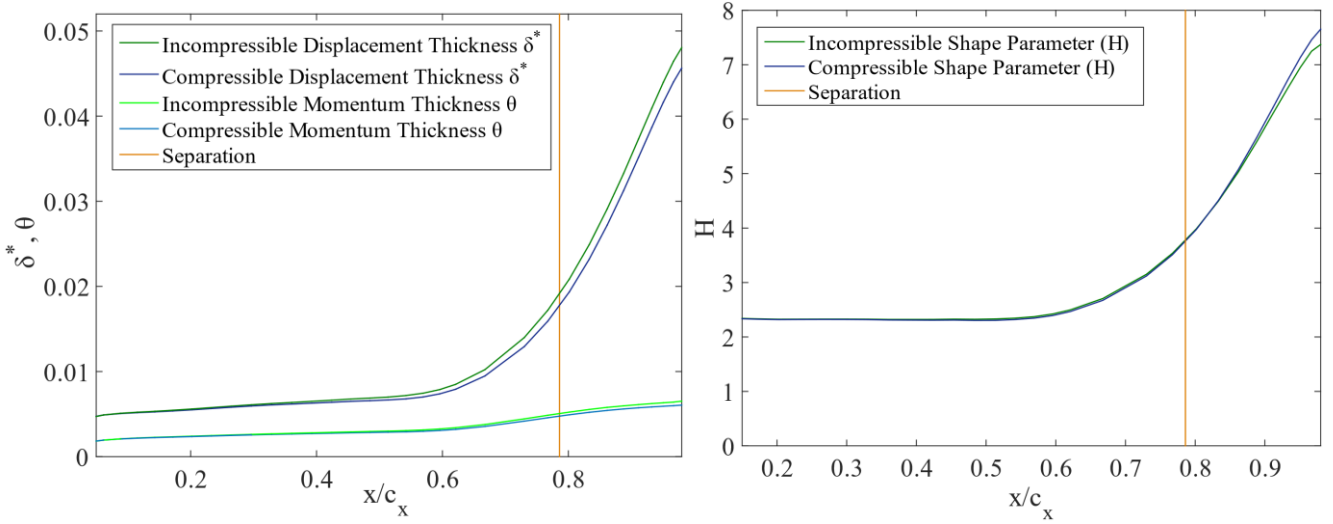


Figure 4.10: Boundary layer parameters comparison between solvers at Reynolds 8,000 and Mach 0.15.

a) Displacement and momentum thickness. b) Shape parameter.

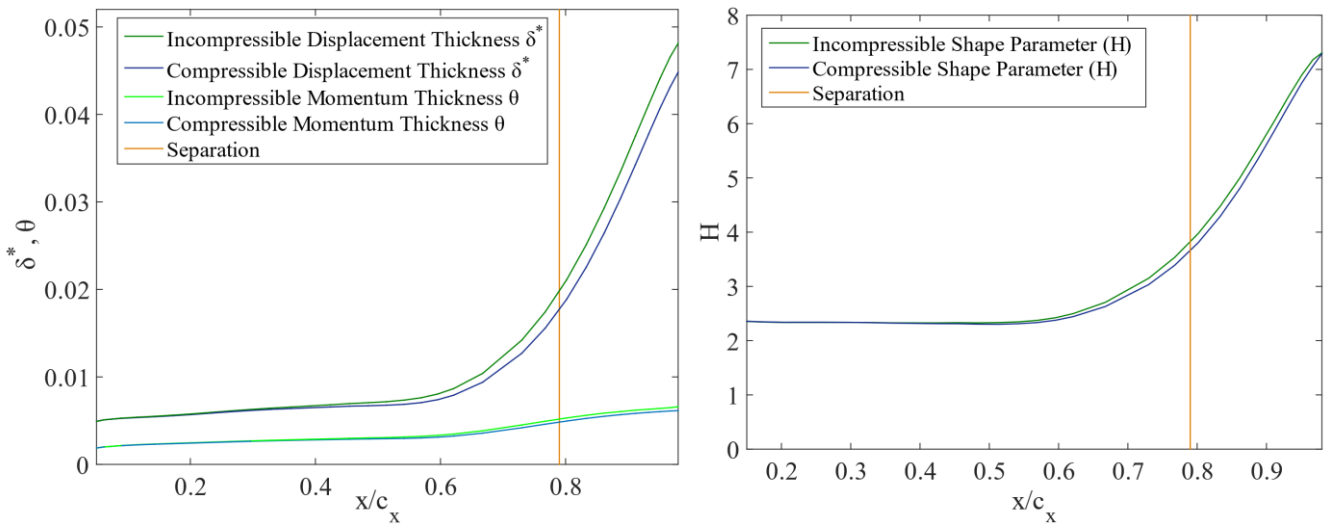


Figure 4.11: Boundary layer parameters comparison between solvers at Reynolds 8,000 and Mach 0.2.

a) Displacement and momentum thickness. b) Shape parameter.

4.5 DNS of Flow Over a T106a Turbine at Reynolds 60,000

The overall conclusion of the analysis performed over the Reynolds number 8,000 case in section 4.4 is that compressibility effects are important and should be considered even with low Mach number cases. The main goal of the thesis was to perform Direct Numerical Simulation of Flow over a T106A low-pressure turbine at a Reynolds number of 60,000 and compare the results given by the Incompressible and Compressible solver. In order to minimise compressibility effects, simulations are performed with a Mach number of 0.1.

The flow field quantities are time averaged over four equivalent chord lengths according to section 3.4.6. Regarding compressibility effects two facts are visualised. On one hand, in Figure 4.12a, the time average Mach number is shown to be restricted to 0.2 in all the flow domain. This implies that the problem could be considered as incompressible, expecting similar results for both solvers. On the other hand, Figure 4.12b displays the percentage of the density with regards to the inlet value. This figure shows how the density in the pressure surface presents the same value as at the inlet, expecting negligible compressibility effects. However, at the suction surface the density is noticeable smaller, especially towards the trailing edge, where we expect compressibility effect, and therefore a diverging solution between the incompressible and compressible results. In Figure 4.12c, the time average artificial viscosity value is visualized. This confirms that once the flow has been initialized with a stabilised flow field, artificial viscosity is no longer required to acquire a stable simulation.

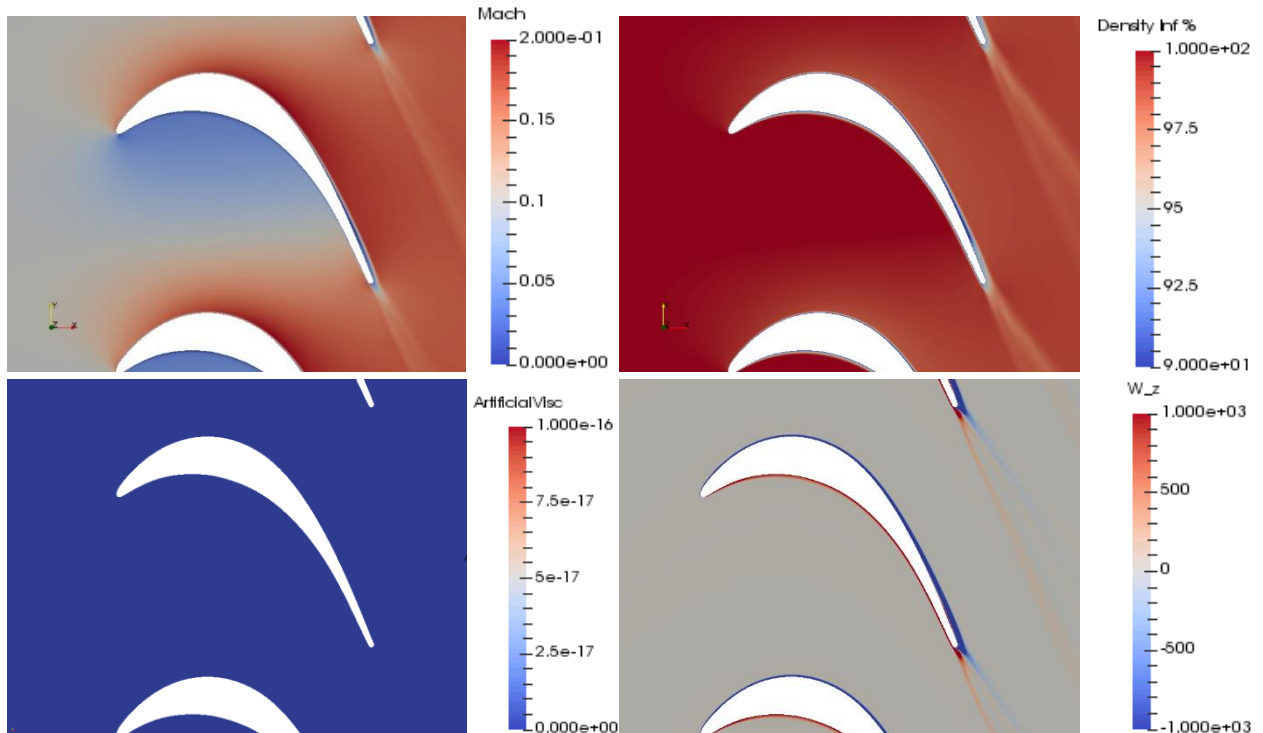


Figure 4.12: Time Averaged quantities for Reynolds number 60,000 at Mach 0.1.

a) Mach number. b) Density. c) Artificial Viscosity. d) Vorticity

4.5.1 Comparison Between Incompressible and Compressible

Firstly, the pressure coefficient results were evaluated. The variance obtained by both solvers is contained within a 10% percentage range. This limit of a 10% variance is identical to the one obtained for the Reynolds 8,000 at Mach number 0.1 case.

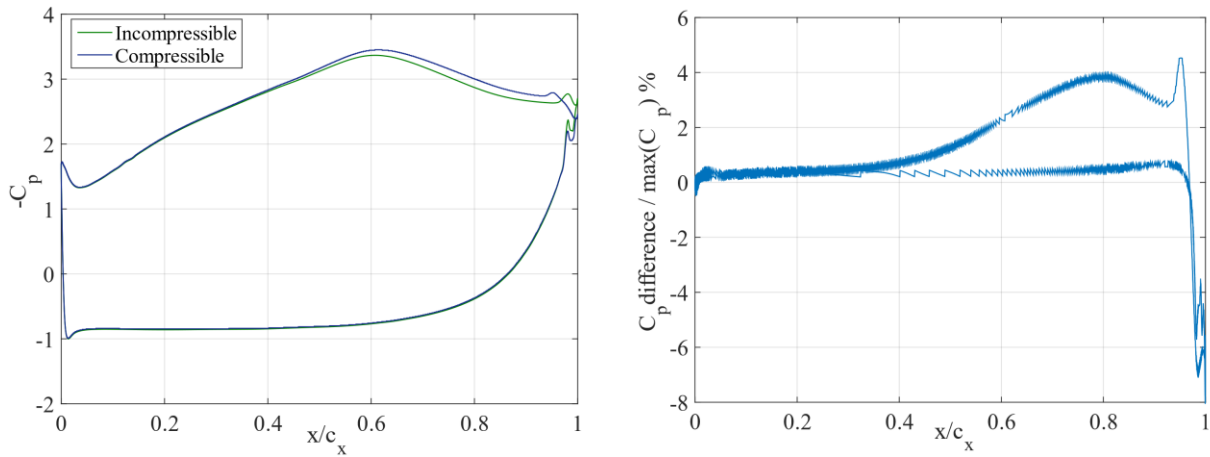


Figure 4.13: Averaged pressure coefficient along the surface of a T106A blade at Reynolds 60,000.

a) Mach number 0.1. b) Difference between compressible and incompressible solvers.

Secondly, the skin friction coefficient obtained by both solvers was compared. The difference in the friction coefficient is greater than the found for the Reynolds 8,000 case. The difference in the location of the separation point triples, exceeding a 2% variance.

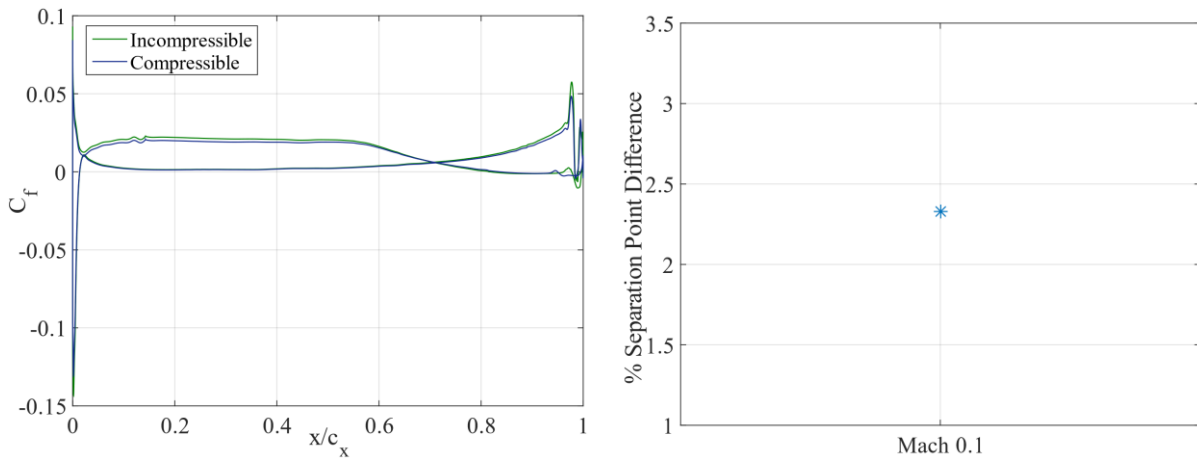


Figure 4.14: Averaged friction coefficient along the surface of a T106A turbine blade at Reynolds 60,000.

a) Mach number 0.1. b) Difference in separation point between compressible and incompressible solvers.

Lastly, the boundary layer is analysed in the suction surface. The result obtained by both solvers agrees close to the leading edge and the difference in the velocity profiles towards the trailing edge is incremented. The velocity profile of the compressible flow solver shows higher values than the incompressible towards the trailing edge, which agrees with the previous results encountered. Separation was found to happen later in the compressible than in the incompressible case. An important fact is that at the trailing edge, the Compressible Flow solver shows positive values of the velocity in all the profile and positive shear stress in a definite. However, this is not observed for the Incompressible case, where the velocity in the trailing edge continues to be reversed. These facts match with a lower displacement thickness and shape parameter for the Compressible case and a steeper decrement of both parameters towards the trailing edge seen in Figure 4.16 and 4.17.

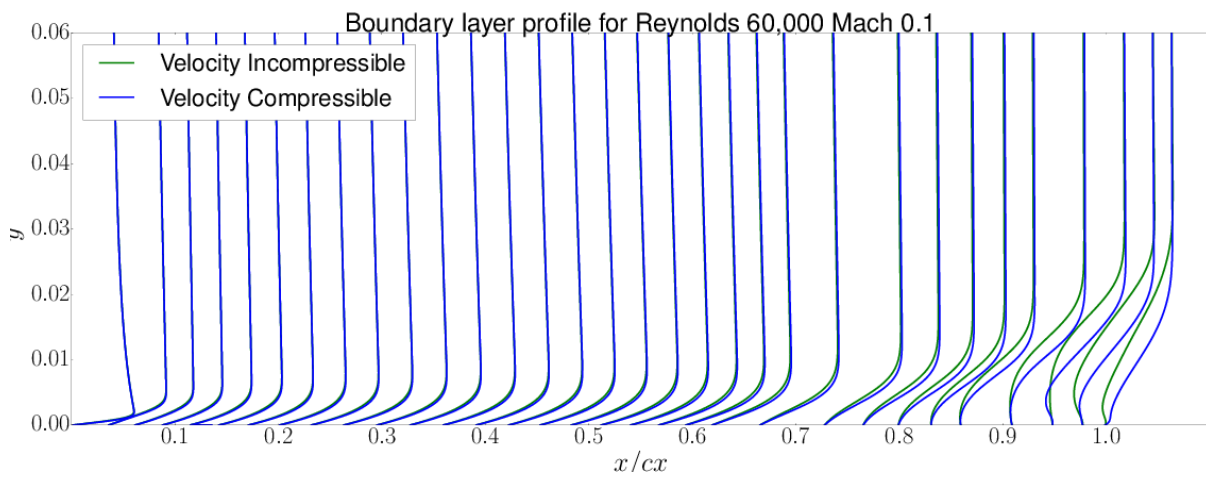


Figure 4.15: Velocity Profile Comparison for Reynolds number 60,000 and Mach 0.1.

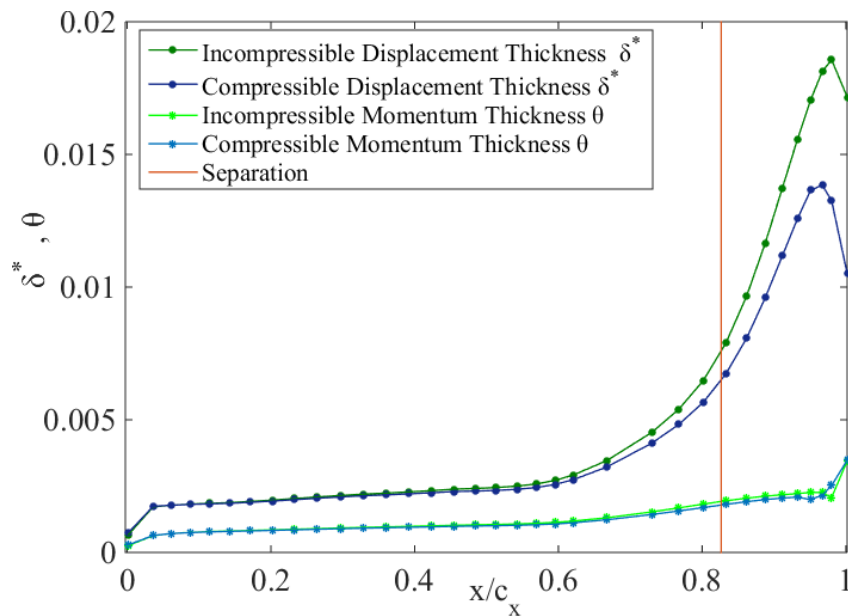


Figure 4.16: Displacement and momentum thickness boundary comparison between solvers at Reynolds 60,000 and Mach 0.1.

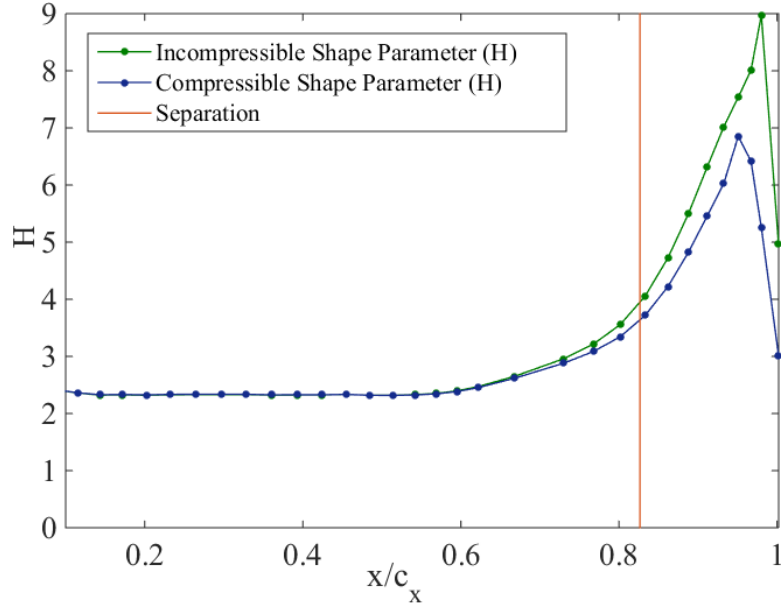


Figure 4.17: Shape parameter comparison between solvers at Reynolds 60,000 and Mach 0.1.

4.6 Time Step Restriction

The time step restriction presented by the Incompressible and Compressible Flow solvers was compared and presented in Figure 4.18. The maximum time step for running a stable simulation was examined for a Reynolds number of 8,000 and 60,000 and the three different Mach number cases of 0.1, 0.15 and 0.2. The time step is expressed in terms of the non-dimensional time step $\Delta t \cdot u_{inlet}$ (the characteristic length is equal to one).

The Compressible solver uses explicit time-integration schemes. The stability of these is governed by the CFL condition and according to section 3.4.3, the time-step restriction follows:

$$\max|\mathbf{v}^e| \frac{c_\lambda}{h} P^2 \Delta t \leq \alpha \quad (4.4)$$

For the Reynolds number 8,000 cases the same mesh and polynomial order are used. According to section 4.4.3 the maximum velocity for all the cases is approximately twice the inlet velocity ($\max|\mathbf{v}^e| \approx 2u_{inlet}$). Being c_λ , α , h and P constants, and knowing $\max|\mathbf{v}^e|$ is approximately twice the inlet velocity, the non-dimensional time-step restriction $\Delta t_{max} \cdot u_{inlet}$ for the Reynolds number 8,000 is expected to be constant, which is confirmed in Figure 4.18-♦.

For the Reynolds number 60,000 different polynomials order have been used to meet DNS requirements for the three Mach cases. Note that in this case, we expect the relation $u_{inlet} \Delta t_{max} P^2$ to be constant, which is shown to be met in Figure 4.18-●.

The Incompressible solver uses a semi-implicit time-integration scheme presenting a much less restrictive CFL condition. For the Reynolds number of 8,000 the Compressible time step is in average 26 times smaller than the Incompressible case. The least difference is 20 times smaller for the Mach 0.2 case and the greatest difference is 36 times smaller for the Mach number

0.1. For the Reynolds number 60,000 the compressible time step is 15 times smaller in average, with the maximum difference being 16 for the Mach number 0.1 and a minimum difference of 14 for the Mach number 0.15 case.

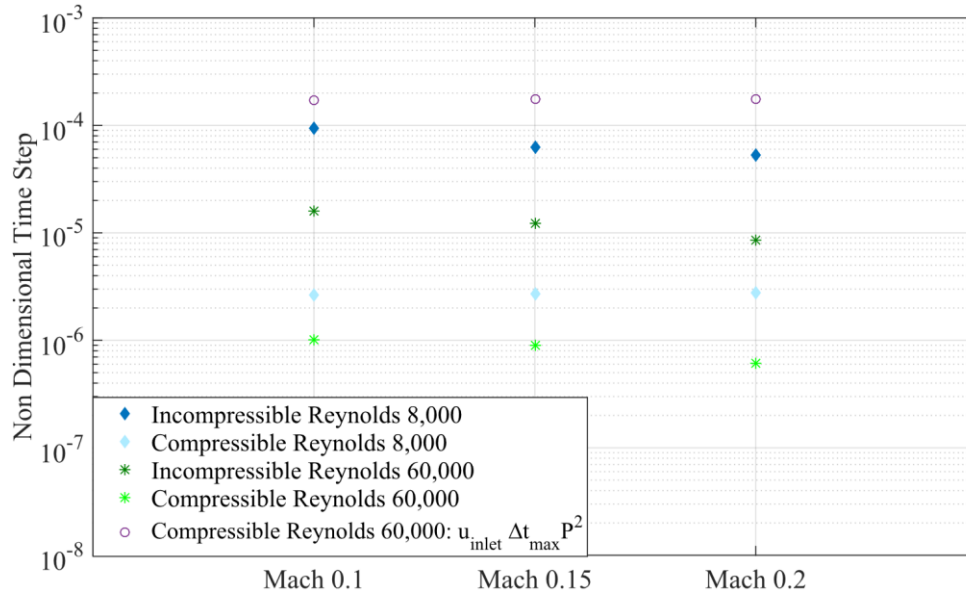


Figure 4.18: Time Step Limitation Comparison between solvers.

4.7 Inlet Boundary Condition Comparison

A critical factor when conducting simulations with the Compressible Flow solver are the specification of the inlet conditions. Initially, equivalent boundary conditions were prescribed for the incompressible and compressible simulations. However, the results obtained presented differences that could not be solely attributed to compressibility effects. Alternatively, the discrepancies were the result of inexact imposition of the inlet boundary conditions in the case of the Compressible Flow solver. Figure 4.19 shows the ratio between the prescribed and the actual inlet boundary conditions for the density, pressure, velocity and inlet angle.

The inequality becomes more significant as the Mach number increases. The same behaviour was visualized at a Reynolds of 60,000, but with an incremented disparity.

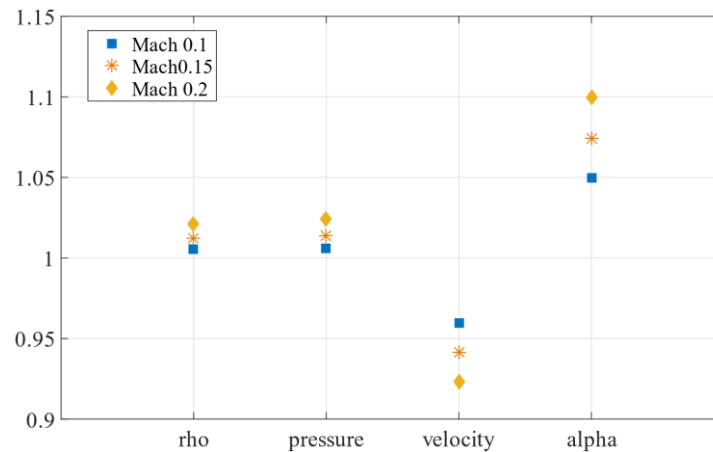


Figure 4.19: Inlet Conditions Discrepancy for the Reynolds number 8,000.

4.8 Scaling of the Parallel Simulation

In this section a study of the scalability of the parallel simulation is conducted over the Reynolds number 60,000 case. This work focuses on a strong scaling approach, in which the problem size stays fixed and the number of CPUs are increased. To assess the scalability, the speed-up S_N and the efficiency E_N are measured. These parameters are calculated as:

$$S_N = \frac{T_1}{T_N} \quad , \quad E_N = \frac{S_N}{N} \quad (4.4)$$

where T_1 is the mean CPU time required to compute a time step when using one CPU and T_N is the mean CPU time when using 'N' number of CPUs.

Figure 4.20 shows the average computational time per time step against the number of CPUs. It shows that increasing the number of processes the computational time decreases, approximately matching the slope of CPU^{-1} . Appreciate that the computational time tends to a constant minimum value for approximately 2500 CPUs. From this point on, the time step barely experiments changes, but the efficiency quickly decreases (see Figure 4.22). The speed-up against the number of CPUs is presented in a logarithm scale axis in Figure 4.21. In this figure, it is better visualised the tendency of the computational time to reduce matching CPU^{-1} . Lastly, Figure 4.22 displays the efficiency against the number of CPUs used. Initially, as the number of CPUs grows, the efficiency increases, achieving a maximum of 1,5 for a number of CPUs close to a thousand.

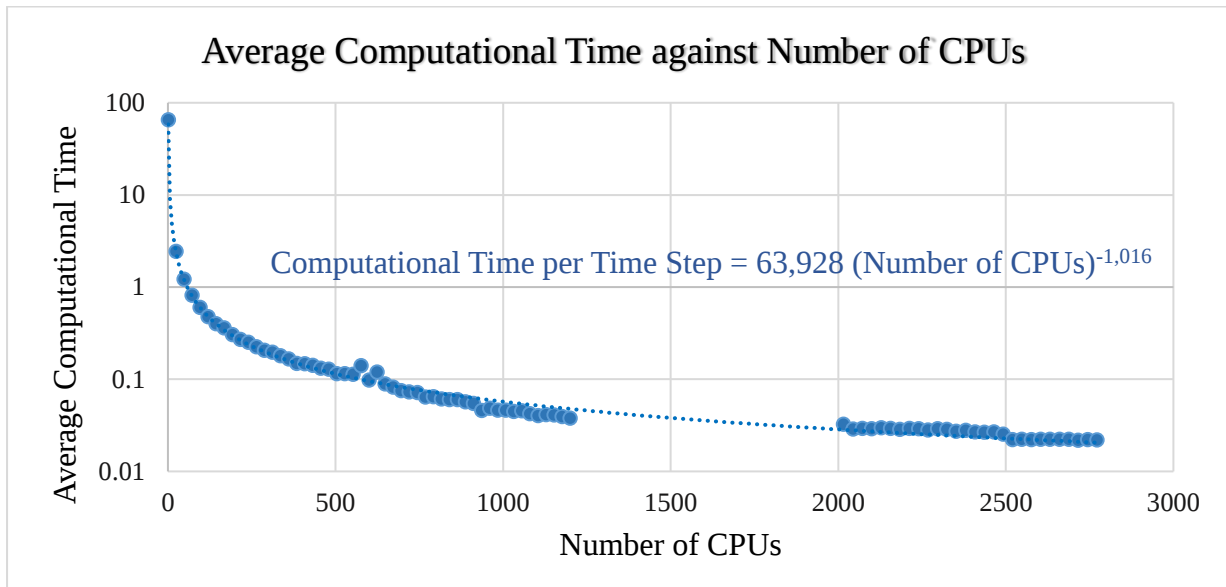


Figure 4.20: Computational time per time-step against the number of CPUs.

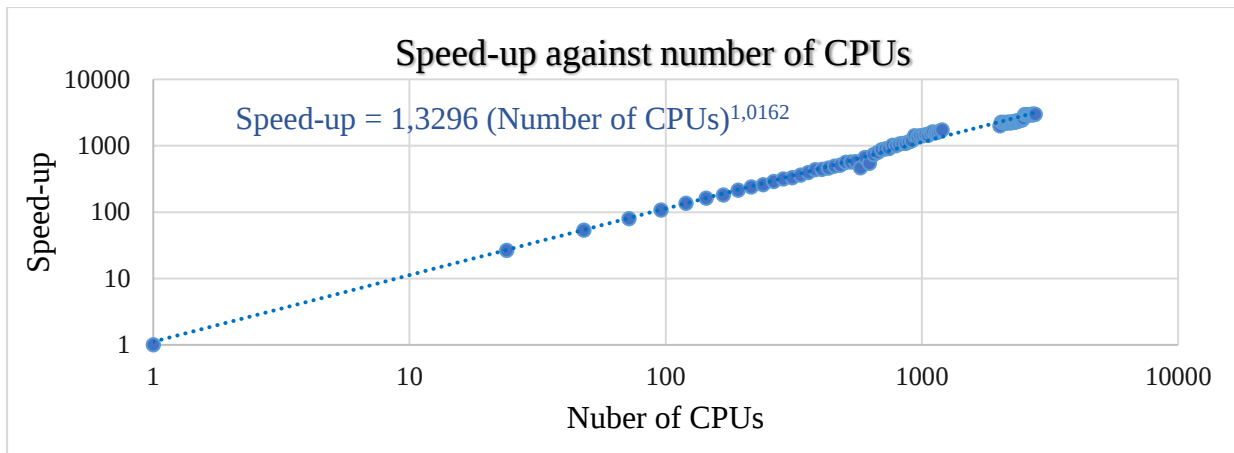


Figure 4.21: Speed-up against the number of CPUs.

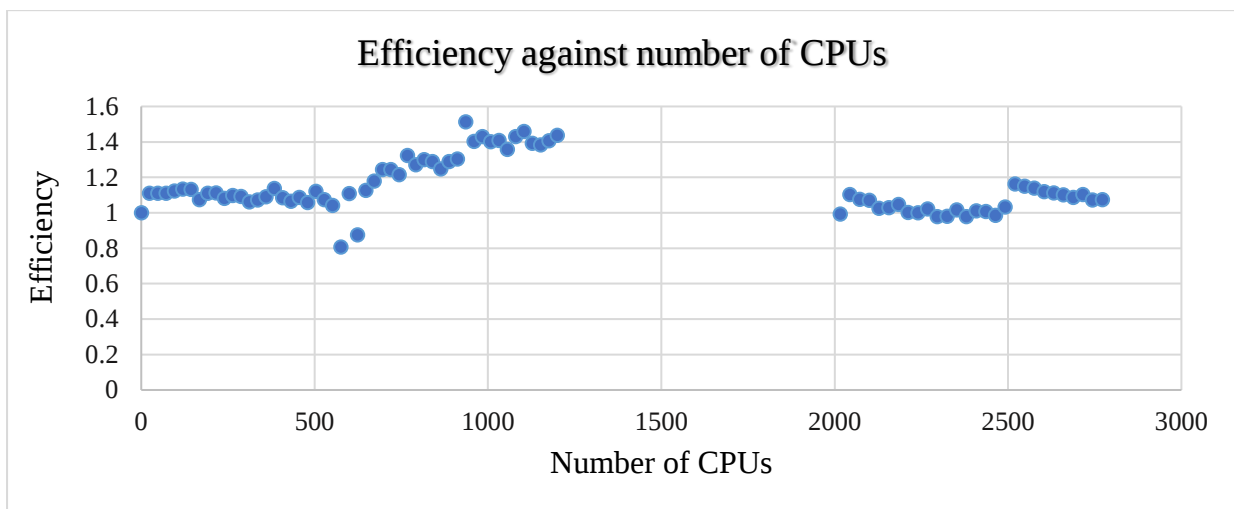


Figure 4.21: Efficiency against the number of CPUs.

Conclusions and Future Work

The Spectral/HP element method has been implemented in the open source Nektar++ to perform DNS on a T106A turbine blade cascade. The flow simulations were performed by solving the compressible Navier-Stokes equations via the Compressible Flow solver. The thesis aims to provide future advice on the use and development of the Compressible Flow solver. The conclusions on the thesis are divided in two sections. First section focuses on the users of the compressible flow solver, with the objective of facilitate the set up and attract people to use the solver. The second section focuses on the development of the code, with the objectives of developing a more stable code and build a more practical solver.

5.1 User Focused Conclusions and Future Work

One of the main downsides of working with the Compressible Flow solver is that Nektar++ user-guide is incomplete and no tutorials were implemented for this solver.

In section 3.3, the importance of the artificial viscosity was highlighted. The smooth artificial model did not run successfully for the cases tested while the non-smooth model resulted to be the optimal solution. However, the user-guide explained how to turn on the smooth artificial viscosity model while the non-smooth model was not explained. The user guide has been updated to illustrate the user how to turn the non-smooth model on. As future work, I would suggest concluding if the smooth artificial viscosity model is also a good solution. If this model fails, the user-guide should be updated to remove the explanation and set up of the smooth artificial model.

In addition, de-aliasing was concluded to have a critical impact on the stability of the solver. As introduced in section 2.8, there are two different strategies, namely a local and a global de-aliasing techniques. The user-guide did not present any section mentioning de-aliasing errors. The user-guide has been updated to introduce the user to both de-aliasing strategies existing. The way to turn them on in Nektar ++ is different. In this thesis, global de-aliasing technique has been proved to give accurate and stable results and therefore its set up has been included in the user-guide.

In order to introduce the user to the spectral/HP element framework *Nektar++* and to describe the main features of its Compressible Flow Solver in a simple manner a tutorial for the compressible Navier-Stokes equations presented in the Compressible Flow solver has been developed and it is attached in appendices. Additionally, it would be beneficial to create a tutorial for the Euler equations.

As a general conclusion, I personally consider of extreme importance to update the user-guide with any new technique developed in the code.

5.2 Code Development Conclusions and Future Work

From Section 4, it is concluded that the differences obtained with the Incompressible and Compressible solvers are a consequence of compressibility effects. This is an important fact as during the project numerous modules have been newly incorporated to the Compressible Flow Solver by the developers. By post-processing the results it was tested that vorticity, wall shear stress and time averaging modules were giving correct results. Moreover, developers updated the time averaging module to include the primitive variables, pressure and velocity, as outputs. This project validated these modules ensuring results converged to the Incompressible results when compressibility effects were negligible.

In section 4.6 the Time Step restriction for the Incompressible and Compressible Flow solver was compared. The Incompressible Flow solver presented a much less restrictive time-step limitation. This limitation was overcome for a moderate Reynolds number of 60,000 but could be critical when simulating higher Reynolds numbers or three-dimensional cases. As future work, I would suggest two developments. Firstly, numerous researches state that the FR_{GU} advection operator presents a more favourable time step restriction. However, this is currently only implemented for segments, quadrilaterals and hexahedra. It would be of interest developing this scheme for further kind of elements. Secondly, in this thesis, for a moderate Reynolds number of 60,000 and two-dimensional simulations, the higher time step limitation of the Compressible solver was overcome. I see essential to compare the stability of both solver under higher Reynolds numbers and three-dimensional simulations and conclude whether the development of a semi-implicit scheme is required for the Compressible Flow solver.

During the development of the thesis numerous cases were tested. Among cases provided to users online, it was found a paper case not currently working for Nektar++ (MMC S11 case [8]). The git repository was analysed to look for a version that managed to run the case provided online. The difference in the versions being artificial viscosity was automatically on. The case only run when artificial viscosity was turned on. As future work, I would analyse the computational cost of having artificial viscosity automatically turned on. This would enhance stability and artificial viscosity would only be added in the case the sensor detects it is required. Moreover, the user is able to see if artificial viscosity is being added as it is included as an output parameter. Thus, the user will decide whether their results are accurate enough or if they require changes as mesh refinement.

Section 4.7 demonstrated the prescribed inlet conditions differ from the actual conditions appearing. This would make the practical use of the compressible flow solver challenging. The fact that the boundary condition is adjusting through the Riemann solver suggests the solid body might need to be further from the inlet and outlet boundaries. Consequently, a possible way to solve this would be extending the domain. However, this would result in the Compressible simulations to be more computationally expensive. There alternative ways to exactly enforce the boundary conditions. An example of it would be to implement the boundary conditions setting the conditions as *USERDEFINETYPE PressureOutflow* in *Nektar++* .xml file. This implementation can be extrapolated for the inlet case.

Finally, in section 4.8, the scalability of the Reynolds 60,000 and Mach 0.1 case was investigated. Showing the tendency of the time step to reduce with the number of CPUS⁻¹, a significant reduction of the averaged time step up to 2500 CPUS and a maximum efficiency obtained for 1000 number of CPUs.

References

- [1] P. Gonzalez, I. Ulizar, and H. . Hodson, “Improved Blade Profiles for High Lift Low Pressure Turbine Applications,” ITP, Industria de Turbo Propulsores, SA, Madrid.
- [2] A. Garai, L. Diosady, S. Murman, and N. Madavan, “DNS of Flow In a Low-Pressure Turbine Cascade Using a Discontinuous-Galerkin Spectral-Element Method,” *ASME Turbo Expo*, pp. 1–10, 2015.
- [3] X. Wu and P. A. Durbin, “Evidence of longitudinal vortices evolved from distorted wakes in a turbine passage,” *J. Fluid Mech. Cambridge University Press*, vol. 446, pp. 199–228, 2001.
- [4] G. Kalitzin, X. Wu, and P. A. Durbin, “DNS of fully turbulent flow in a LPT passage,” *International Journal of Heat Fluid Flow*, vol. 24, no. 4, pp. 636–644, 2003.
- [5] T. A. ZAKI, J. G. WISSINK, W. RODI, and P. A. DURBIN, “Direct numerical simulations of transition in a compressor cascade: the influence of free-stream turbulence,” *J. Fluid Mech, Cambridge University Press* vol. 665, pp. 57–98, 2010.
- [6] M. M. Rai, “Direct Numerical Simulations of the Transitional/Turbulent Wake of a Flat Plate,” *Journal of Propulsion and Power*, vol. 26, no. 3, pp. 575–586, 2010.
- [7] G. Karniadakis and S. Sherwin, “Spectral/hp Element Methods for Computational Fluid Dynamics,” in *Spectral/hp Element Methods for Computational Fluid Dynamics*, 2nd ed., Oxford University Press, 2005.
- [8] C. Cantwell, D. Moxey, A. Comerford, A. Bolis, G. Rocco, G. Mengaldo, D. De Grazia, S. Yakovlev, J.-E. Lombard, D. Ekelschot, B. Jordi, H. Xu, Y. Mohamied, C. Eskilsson, B. Nelson, P. Vos, C. Biotto, R.M. Kirby, S.J. Sherwin, “Nektar++: An open-source spectral/hp element framework,” *Compututer Physics Communications*, vol. 192, pp. 205–219, Elsevier 2015.
- [9] D. De Grazia, “Three-dimensional discontinuous spectral/hp element methods for compressible flows,” Department of Aeronautics, Imperial College London, 2016.
- [10] G. Mengaldo, D. De Grazia, J. Peiro, A. Farrington, F. Witherden, and S. J. Sherwin, “A Guide to the Implementation of Boundary Conditions in Compact High-Order Methods for Compressible Aerodynamics,” *Aiaa 2014-2923*, no. June, pp. 1–33, 2014.
- [11] S. K. Godunov, “A finite difference method for the computation of discontinuous

- solutions of the equations of fluid dynamics.,” pp. 652–656, 1959.
- [12] G. Mengaldo, “Discontinuous spectral/hp element methods: development, analysis and applications to compressible flows,” Department of Aeronautics, Imperial College London, 2015.
 - [13] Q. Abbas and J. Nordström, “Weak Versus Strong No-Slip Boundary Conditions for the Navier-Stokes Equations,” *Eng. Appl. Comput. Fluid Mech.*, vol. 1, no. 4, pp. 29–38, 2010.
 - [14] J. Nordström, S. Eriksson, and P. Eliasson, "Weak and strong wall boundary procedures and convergence to steady-state of the Navier-Stokes equations", *Journal of Computational Physics*, vol. 231, no. 14, pp. 4867–4884, 2012.
 - [15] G. Mengaldo, D. De Grazia, D. Moxey, P. E. Vincent, and S. J. Sherwin, “Dealiasing techniques for high-order spectral element methods on regular and irregular grids,” *Journal of Computational Physics.*, 2015.
 - [16] User Guide, Version 4.4.0 “Nektar ++ : Spectral / hp Element Framework,” March, 2017.
 - [17] Stadtmüller, P., 2001. "Investigation of wake-induced transition on the LP turbine cascade T106A-EIZ". In DFGVerbundprojekt Fo 136//11, Version 1.0, University of the Armed Forces, Munich, Germany.
 - [18] Stieger, R. D., and Hodson, H. P., 2003. "The transition mechanism of highly-loaded LP turbine blades". ASME Turbo Expo Paper GT2003-38304.
 - [19] J. G. Wissink, “DNS of separating, low Reynolds number flow in a turbine cascade with incoming wakes,” *Int. J. Heat Fluid Flow*, vol. 24, no. 4, pp. 626–635, 2003.
 - [20] V. Michelassi, L. Chen, R. Pichler, and R. D. Sandberg, “GT2014-25689,” 2014.
 - [21] L. T. Diosady, S. M. Murman, and N. K. Madavan, “DNS of Low-Pressure Turbine Cascade Flows With Elevated Inflow Turbulence Using a Discontinuous-Galerkin Spectral-Element Method,” *ASME Turbo Expo*, pp. 1–16, 2016.
 - [22] R. D. Stieger, “The Effects of Wakes on Separating Boundary Layers in Low Pressure Turbines,” Cambridge University, 2002.
 - [23] N. Georgiadis, D. Rizzetta, and C. Fureby, “Large-Eddy Simulation: Current Capabilities, Recommended Practices, and Future Research,” *47th AIAA Aerospace*

Science Meeting. Incl. New Horizons Forum Aerosp. Expo., no. July, 2009.

- [24] J. Stoker, “Direct Numerical Simulation of Flow Over a T106A Low Pressure Turbine at $Re=50K$,” Department of Aeronautics, Imperial College London, 2015.
- [25] R. Courant, K. Friedrichs, and H. Lewy, “On the Partial Difference Equations of Physics,” *New York*, no. March, p. 92, 1956.
- [26] P. Fernandez, N. C. Nguyen, and J. Peraire, “The hybridized Discontinuous Galerkin method for Implicit Large-Eddy Simulation of transitional turbulent flows,” *Journal of Computational Physics.*, vol. 336, pp. 308–329, 2017.
- [27] A. Uranga, P.-O. Persson, M. Drela, and J. Peraire, “Implicit Large Eddy Simulation of transition to turbulence at low Reynolds numbers using a Discontinuous Galerkin method,” *International Journal for Numerical Methods in Engineering*, 28 October, 2010.

Compressible Flow Solver Tutorial

A2.1 Goals

The aim of this tutorial is to introduce the user to the spectral/*hp* element framework *Nektar++* and to describe the main features of its Compressible Flow Solver in a simple manner. After the completion of the tutorial the user will be familiar with:

- The setup of the initial and boundary conditions, the parameters and the solver settings;
- The expansions set up to mitigate aliasing effects;
- The addition of artificial viscosity to deal with flow discontinuities and the consequential numerical oscillations;
- Running a simulation with the Compressible Flow Solver;
- The importance of running the simulation in parallel with multiple processors.
- The post-processing of the data and the visualisation of the results in Paraview or VisIt;
- The creation of Paraview animation to monitor the evolution of the simulation or visualize non-steady simulations; and
- The use of FieldConvert modules to extract useful quantities from the field variables.

A2.2 Problem Selection

The problem was selected in terms of the size of the completed tutorial folder, its stability and the computational time required to complete the simulation. An additional fact when selecting the tutorial case was that the problem is part of the regression tests *Nektar++* verifies when it is installed. The possibilities of the tutorial case failing due to new versions of the code minimises while there is no need to include a regression test for the tutorial case. Moreover, this case allows to visualize the flow evolution with few snapshots of the flow. This fact facilitated the creation of a video in Paraview.

When selecting the tutorial case, the total memory of the tutorial folder was an important asset. The size of the folder containing the Compressible Solver Tutorial has been carefully considered to be below the average size of the rest of tutorial folders. As shown in table X, the average size is 10.16 MiB while the size of the tutorial created is 6.5 MiB.

Tutorials	Case study	Folder Size
Basics	Advection-Diffusion	0.725 MiB
Basics	Helmholtz	0.395 MiB
Fundamentals	Integration	0.112 MiB
Fundamentals	Differentiation	0.127 MiB
Incompressible Navier-Stokes	Taylor-Green vortex	44.4 MiB
Global Flow Stability Computation	Channel flow	1.2 MiB
Global Flow Stability Computation	Cylinder flow	29.1 MiB
Global Flow Stability Computation	Backward-facing step	5.3 MiB
Average size		10.16 MiB

Table A.1.1: Total folder size of the different tutorial cases.

The stability of the tutorial case was examined to provide a relatively large time step. As explained in the thesis the non-dimensional wall distance is inversely proportional to the Reynolds number. In order to allow for a coarse mesh, a low Reynolds number of 200 has been imposed. A coarse mesh and a low Mach number of 0.2 provides a CFL condition less restrictive and therefore a higher time step.

The dimensionless time step of the compressible flow tutorial is $6.8e-4$, which is larger than the average value. This was expected as the Compressible Flow Solver uses an explicit scheme, which restricts the stability of the solver.

Tutorials	Case study	Dimensionless Time Step
Basics	Advection-Diffusion	1.10^{-3}
Incompressible Navier-Stokes	Taylor-Green vortex	5.10^{-3}
Global Flow Stability Computation	Channel flow	1.10^{-3}
Global Flow Stability Computation	Cylinder flow	1.10^{-3}
Global Flow Stability Computation	Cylinder flow Receptivity	8.10^{-4}
Global Flow Stability Computation	Backward-facing step	2.10^{-3}
Average Time Step		$1.8.10^{-3}$

Table A.2.2: Comparison of the time step for the different tutorials.

The tutorial consists of two simulations. The first shows how to set and run the Solver.

Steps	Processors	Time
1000	1	3
1000	32	8

Table A.2.3: Computational Time to perform the first part of the tutorial.

The second simulation uses the same parameters but runs longer to create the .chk files required to create the video and visualize the field. This second case, shows the user importance of running the simulation in parallel.

Final Physical Time	Processors	Time
0.6	4	1h
0.6	32	8min

Table A.2.4: Computational Time to perform the second part of the tutorial.



Compressible Flow Solver: Navier Stokes equations

Tutorials

July 21, 2017

Department of Aeronautics, Imperial College London, UK
Scientific Computing and Imaging Institute, University of Utah, USA

Introduction

The aim of this tutorial is to introduce the user to the spectral/*hp* element framework *Nektar++* and to describe the main features of its Compressible Flow Solver in a simple manner. If you have not already downloaded and installed *Nektar++*, please do so by visiting www.nektar.info, where you can also find the [User-Guide](#) with the instructions to install the library.

This tutorial requires:

- *Nektar++* CompressibleFlowSolver and pre- and post-processing tools,
- The visualisation tool [Paraview](#) or [VisIt](#)

1.1 Goals

After the completion of this tutorial, you will be familiar with:

- The setup of the initial and boundary conditions, the parameters and the solver settings;
- The expansions set up to mitigate aliasing effects;
- The addition of artificial viscosity to deal with flow discontinuities and the consequential numerical oscillations;
- Running a simulation with the CompressibleFlow solver;
- The post-processing of the data and the visualisation of the results in Paraview or VisIt;
- The creation of Paraview animation to monitor the evolution of the simulation or visualize non-steady simulations; and
- The use of FieldConvert modules to extract useful quantities from the field variables.

**Task 1.1**

Prepare for the tutorial. Make sure that you have:

- Installed and tested *Nektar++* v4.5.0 from a binary package, or compiled it from source. By default binary packages will install all executables in `/usr/bin`. If you compile from source they will be in the sub-directory `dist/bin` of the `build` directory you created in the *Nektar++* source tree. We will refer to the directory containing the executables as `$NEK` for the remainder of the tutorial.
- Downloaded the tutorial files: http://doc.nektar.info/tutorials/4.5.0/cfs/CylinderSubsonic_NS/cfs-CylinderSubsonic_NS.tar.gz. Unpack it using `unzip cfs-CylinderSubsonic_NS.tar.gz` to produce a directory `cfs-CylinderSubsonic_NS` with subdirectories called `tutorial` and `complete`. We will refer to the `tutorial` directory as `$NEKTUTORIAL`.

**Task 1.2**

Additionally, you should also install

- a visualization package capable of reading VTK files, such as ParaView (which can be downloaded from [here](#)) or VisIt (downloaded from [here](#)). Alternatively, you can generate Tecplot formatted `.dat` files for use with Tecplot.

1.2 Background

The Compressible Flow Solver allows us to solve the unsteady compressible Euler and Navier-Stokes equations for 1D/2D/3D problems using a discontinuous representation of the variables. For a more detailed description of this solver, please refer to the [User-Guide](#).

In this tutorial we focus on the 2D Compressible Navier-Stokes equations. The two-dimensional second order partial differential equations can be written as:

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} = 0, \quad (1.1)$$

where $\mathbf{q}$ is the vector of the conserved variables,

$$\mathbf{q} = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{Bmatrix} \quad (1.2)$$

where ρ is the density, u and v are the velocity components in x and y directions, p is the pressure and E is the total energy. In this work we considered a perfect gas law for which the pressure is related to the total energy by the following expression:

$$E = \frac{p}{\gamma - 1} + \frac{1}{2}\rho(u^2 + v^2), \quad (1.3)$$

where γ is the ratio of specific heats.

The vector of the fluxes $\mathbf{f} = \mathbf{f}(\mathbf{q}, \nabla(\mathbf{q}))$ and $\mathbf{g} = \mathbf{g}(\mathbf{q}, \nabla(\mathbf{q}))$ can also be written as:

$$\mathbf{f} = \mathbf{f}_i - \mathbf{f}_v, \quad \mathbf{g} = \mathbf{g}_i - \mathbf{g}_v, \quad (1.4)$$

The inviscid fluxes $\mathbf{f}_i$ and $\mathbf{g}_i$ take the form:

$$\mathbf{f}_i = \begin{Bmatrix} \rho u \\ p + \rho u^2 \\ \rho uv \\ u(E + p) \end{Bmatrix}, \quad \mathbf{g}_i = \begin{Bmatrix} \rho v \\ \rho uv \\ p + \rho v^2 \\ v(E + p) \end{Bmatrix}, \quad (1.5)$$

while the viscous fluxes $\mathbf{f}_v$ and $\mathbf{g}_v$ take the following form:

$$\mathbf{f}_v = \begin{Bmatrix} 0 \\ \tau_{xx} \\ \tau_{yx} \\ u\tau_{xx} + v\tau_{yx} + kT_x \end{Bmatrix}, \quad \mathbf{g}_v = \begin{Bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ u\tau_{xy} + v\tau_{yy} + kT_y \end{Bmatrix}, \quad (1.6)$$

where τ_{xx} , τ_{xy} , τ_{yx} and τ_{yy} , are the components of the stress tensor¹

$$\begin{aligned} \tau_{xx} &= 2\mu \left(u_x - \frac{u_x + v_y}{3} \right), \\ \tau_{yy} &= 2\mu \left(v_y - \frac{u_x + v_y}{3} \right), \\ \tau_{xy} &= \tau_{yx} = \mu(v_x + u_y), \end{aligned} \quad (1.7)$$

where μ is the dynamic viscosity calculated using the Sutherland's law and k is the thermal conductivity.

¹Note that we use Stokes hypothesis $\lambda = -2/3$.

1.3 Problem description

We aim to simulate the flow past a cylinder by solving the Compressible Navier Stokes equations. For our study we use the following free-stream parameters: A Mach number equal to $M_\infty = 0.2$, a Reynolds number $Re_{L=1} = 200$ and $Pr = 0.72$, with the pressure set to $p_\infty = 101325 \text{ Pa}$ and the density equal to $\rho = 1.225 \text{ Kg/m}^3$.

The flow domain is a rectangle of sizes $[-10 \ 20] \times [-10 \ 10]$. The mesh consists of 639 quadrilaterals in which we applied the following boundary conditions (BCs): *Non – slip isothermal wall* on the cylinder surface, *far – field* at the bottom and top boundaries, *inflow* at the left boundary and *outflow* at the right boundary.

For the Navier-Stokes equations a *non – slip* condition must be applied to the velocity field at a solid wall, which corresponds to the cylinder for this problem. The cylinder wall is defined as an isothermal wall with imposed temperature $T_{wall} = 300.15 \text{ K}$.

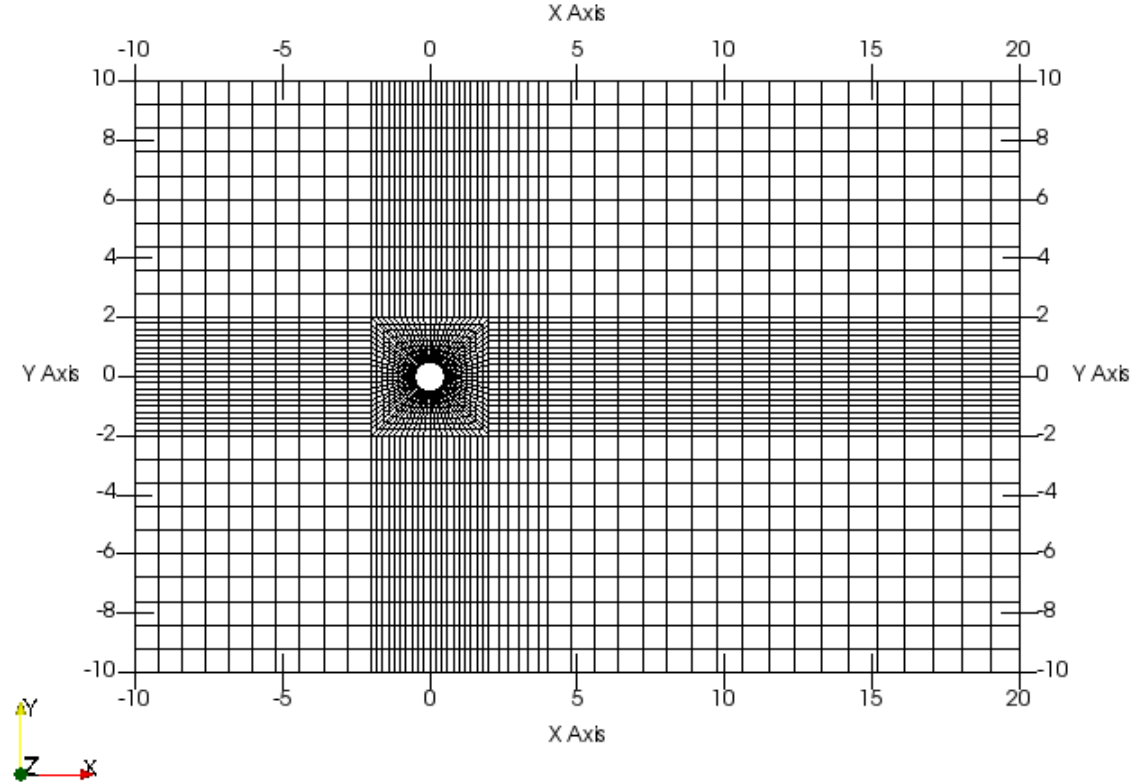


Figure 1.1 639 elements mesh.

Inflow, Outflow and Farfield BCs:

In the Compressible Flow Solver the boundary conditions are weakly implemented- (i.e the BCs are applied to the fluxes). In the Euler equations, for farfield BCs, the flux is

computed via a Riemann solver. The use of a Riemann solver for applying BCs implies the usage of a ghost point where it is necessary to apply a consistent ghost state, which is not always trivial. In evaluating the boundary, the Riemann solver takes automatically into account the eigenvalues (characteristic lines) of the Euler equations and therefore the problem is always well posed. This approach is equivalent to a characteristic approach where the corresponding Riemann invariants are computed and applied as BCs, taking into account if the boundary is an inflow or outflow. The method is also known as no-reflective BCs as it damps the spurious reflections from the boundaries.

The characteristic approach presented for the Euler equations for farfield boundaries, works also for the advective flux of the Navier-Stokes equations in regions where viscosity effects can be neglected. However, in our outflow case shedding is present, so viscosity effects become important. In this case, the characteristic treatment of the BCs generates spurious oscillations polluting the overall solution and leading to numerical instabilities. In order to avoid this, *Nektar++* implements a method based on the so-called sponge terms, modifying the RHS of the compressible NS equations as follows:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_1}{\partial x_1} + \frac{\partial \mathbf{f}_2}{\partial x_2} = \sigma(\bar{\mathbf{x}})(\mathbf{u}_{\text{ref}} - \mathbf{u}), \quad (1.8)$$

Where $\sigma(\bar{\mathbf{x}})$ is a damping coefficient defined in a region $\bar{\mathbf{x}}$ in proximity to the boundaries and u_{ref} is a known reference solution. The length and the shape of the damping coefficient depend on the problem being solved.

For further understanding of the boundary conditions implementation, please visit [A Guide to the Implementation of Boundary Conditions in Compact High-Order Methods for Compressible Aerodynamics](#).

The initial condition is chosen to be that of a free flow field without the cylinder. If the solution greatly differs from the initial condition waves develop giving stability problems.

Tip

Set the initial conditions close to the expected solution to accelerate convergence and increment stability. Examples of setting more realistic initial conditions:



- In the case of a low Mach number, an incompressible flow solution can be used as initial condition.
- Also, setting an inviscid solution as initial conditions may help. Note that this can be done by selecting Euler equations instead of Navier-Stokes in the `SOLVERINFO` tag.

We successively setup the parameters of the problem (section 2.3). We finally run the solver (section 3) and post-process the data in order to visualise the results (section 4).

Pre-processing

To set up the problem we have three steps. The first is setting up a mesh as discussed in section 2.1. The second one is setting the expansion bases as explained in section 2.2. We also need to configure the problem initial conditions, boundary conditions and parameters which are discussed in 2.3.

2.1 Mesh generation

The first pre-processing step consists in generating the mesh in a *Nektar++* compatible format. One option to do this is to use the open-source mesh-generator Gmsh to first create the geometry. The mesh format provided by Gmsh is not consistent with the *Nektar++* solvers and, therefore, it needs to be converted. An example of how to do this can be found in the [Advection Solver Tutorial](#).

For two-dimensional simulations, the mesh definition contains 6 tags encapsulated within the `GEOMETRY` tag. The first tag, `VERTEX`, contains the spatial coordinates of the vertices of the various elements of the mesh. The second tag, `EDGE` contains the lines connecting the vertices. The third tag, `ELEMENT`, defines the elements (note that in this case we have only quadrilateral - e.g. `<Q ID="85">` - elements). The fourth tag, `CURVED`, is used to describe the control points for the curve. Note this tag is only necessary if curved edges or faces are present in the mesh and may otherwise be omitted. The fifth tag, `COMPOSITE`, is constituted by the physical regions of the mesh called **composite**, where the composites formed by elements represent the solution sub-domains - i.e. the mesh sub-domains where we want to solve our set of equations (note that we will use these composites to define expansion bases on each sub-domain in section 2.3) - while the composites formed by edges are the boundaries of the domain where we need to apply suitable boundary conditions (note that we will use these composites to specify the boundary conditions in section 2.3). Finally, the sixth tag, `DOMAIN`, formally specifies the overall solution domain as the union of the composites forming the solution subdomains (note that the specification of different subdomain - i.e. composites - in this case is not necessary since they are constituted by same element shapes). For additional details on

the `GEOMETRY` tag refer to the [User-Guide](#).

```

1 <?xml version="1.0" encoding="utf-8" ?>
2 <NEKTAR>
3   <GEOMETRY DIM="2" SPACE="2">
4     <VERTEX>
5       <V ID="0">-1.00000000e+01 1.00000000e+01 0.00000000e+00 </V>
6       ...
7       <V ID="706">-4.93844170e-01 -7.82172325e-02 0.00000000e+00 </V>
8     </VERTEX>
9     <EDGE>
10      <E ID="0"> 0 1 </E>
11      ...
12      <E ID="1346"> 706 668 </E>
13    </EDGE>
14    <ELEMENT>
15      <Q ID="0"> 0 1 2 3 </Q>
16      ...
17      <Q ID="639"> 1345 1346 1269 615 </Q>
18    </ELEMENT>
19    <CURVED>
20      <E ID="0" EDGEID="1344" NUMPOINTS="4" TYPE="PolyEvenlySpaced"> ...
21      ...
22      <E ID="1346" EDGEID="235" NUMPOINTS="4" TYPE="PolyEvenlySpaced"> ...
23    </CURVED>
24    <COMPOSITE>
25      <C ID="100"> E[1268,1271,...,1344,1346] </C>
26      <C ID="101"> E[3,6,...,1256,1266] </C>
27      ...
28      <C ID="0"> Q[0-639] </C>
29    </COMPOSITE>
30    <DOMAIN> C[0] </DOMAIN>
31  </GEOMETRY>
32 </NEKTAR>

```

Note



In this case the mesh has been defined under the `GEOMETRY` tag with the `EXPANSIONS` definition and the `CONDITIONS` section in the same .xml file. However, the mesh can be a separate input .xml format containing only the geometry definition. Also, note that this mesh is in uncompressed format. In order to reduce the size of a large mesh compressed format should be used.

2.2 Expansion bases

We need to specify the expansion bases we want to use in each of the composites or sub-domains (`COMPOSITE=".."`) introduced in section 2.1:

```

1 <EXPANSIONS>
2   <E COMPOSITE="C[0]" NUMMODES="3" FIELDS="rho,rhou,rhov,E"
3     TYPE="MODIFIED" />
4 </EXPANSIONS>

```

For this case there is only one composite, `COMPOSITE="C[0]"`, where `NUMMODES` is the number of coefficients we want to use for the basis functions (that is commonly equal to $P+1$ where P is the polynomial order of the basis functions), `TYPE` allows selecting the basis functions, `FIELDS` is the solution variable of our problem and `COMPOSITE` are the mesh regions. For additional details on the `EXPANSIONS` tag refer to the [User-Guide](#).

Tip



One source of instability is aliasing effects which arise from the nonlinearity of the underlying problem. Dealiasing techniques based on the concept of consistent integration can be applied in order to improve the robustness of the solver. For further information about dealiasing techniques, please check [Dealiasing techniques for high-order spectral element methods on regular and irregular grids](#).

An example of dealiasing technique on for quadrilateral elements:

```
1 <EXPANSIONS>
2   <E COMPOSITE="C[0]" BASISTYPE="GLL_Lagrange, GLL_Lagrange"
3     NUMMODES="5,5" POINTSTYPE="GaussLobattoLegendre, GaussLobattoLegendre"
4     NUMPOINTS="10,10" FIELDS="rho, rho_u, rho_v, E" />
5 </EXPANSIONS>
```

2.3 Configuring problem definitions

We will now proceed to set up the various problem parameters, the solver settings, initial and boundary conditions.

Parameters

The case will be run at Mach number equal to $M_\infty = 0.2$, Reynolds number $Re_{L=1} = 200$ and $Pr = 0.72$, with the pressure set to $p_\infty = 101325 \text{ Pa}$ and the density equal to $\rho = 1.225 \text{ Kg/m}^3$. The cylinder is defined as an isothermal wall with imposed temperature $T_{wall} = 300.15 \text{ K}$.

Within `PARAMETERS` tag, we can also define the final physical time of the simulation, `FinTime`, the number of steps `NumSteps`, the step-interval when an output file is written `I0_CheckSteps` and the step-interval when information about the simulation is printed to the screen `I0_InfoSteps`.



Task 2.1

In the .xml file under the tag `PARAMETERS`, define all the flow parameters as described above. These are declared as $Mach$, Re , Pr , p_{inf} , ρ_{hinf} and T_{wall} . Define the number of steps $NumSteps$ as the ratio of the $FinalTime$ to the time-step $TimeStep$.

**Warning**

Do not define both *Prandtl number* and the *thermal conductivity* parameters. They are correlated and defining both will prevent the simulation to start.

```

1 <PARAMETERS>
2   <P> TimeStep = 0.00001 </P>
3   <P> FinTime = 0.01 </P>
4   <P> NumSteps = FinTime/TimeStep </P>
5   <P> IO_CheckSteps = 100 </P>
6   <P> IO_InfoSteps = 100 </P>
7   <P> GasConstant = 287.058 </P>
8   <P> Gamma = 1.4 </P>
9   <P> plnf = 101325 </P>
10  <P> rhoInf = 1.225 </P>
11  <P> Mach = 0.2 </P>
12  <P> clnf = sqrt(Gamma * plnf / rhoInf) </P>
13  <P> ulnf = Mach*clnf </P>
14  <P> vlnf = 0.0 </P>
15  <P> Twall = 300.15 </P>
16  <P> Re = 200 </P>
17  <P> L = 1 </P>
18  <P> mu = rhoInf * L * ulnf / Re </P>
19  <P> Pr = 0.72 </P>
20 </PARAMETERS>

```

Solver Settings

We now declare how the flow will be solved. We want to include the effects of fluid viscosity and heat conduction and consequently the equation type we are going to use is the Navier-Stokes equations.

Note

In *Nektar ++* the spatial discretization of the compressible Navier-Stokes equations is projected in the polynomial space via discontinuous projection. Specifically we make use of either of the discontinuous Galerkin (DG) method or the Flux-Reconstruction (FR) approach. Consequently, set the `Projection` to `DisContinuous`, as Continuous Projection is not supported in the Compressible Flow Solver.

We must specify the advection type which will be the classical DG in weak form. Note *Nektar ++* also presents the FR_{DG} scheme, which recovers the DG scheme with exact mass matrix, the FR_{HU} scheme, which recovers the DG scheme with lumped mass matrix and the FR_{SD} scheme, which recovers a spectral difference scheme. We must also define the diffusion operator we want to use, which will be local Discontinuous Galerkin and the time integration method which will be the Classical Runge Kutta of order 4.

Tip

When selecting the Advection Type scheme, bear in mind that:



- The error associated with the FR_{DG} and $DG_{SEM} - EMM$ scheme is the lowest. It corresponds to the most accurate scheme but it also presents the most severe restrictions in terms of time-step.
- The FR_{HU} and FR_{SD} are slightly less accurate but have more favourable time-step restrictions.
- For further understanding, please visit [Connections between the discontinuous Galerkin method and high-order flux reconstruction schemes](#) and [On the Connections Between Discontinuous Galerkin and Flux Reconstruction Schemes: Extension to Curvilinear Meshes.](#)

Additionally, we need to define the Upwind Type (i.e. Riemann solver) we want to use for the advection operator. For this problem we will use HLLC (Harten, Lax, van Leer+Contact) Riemann solver. Also, we will use the constant viscosity type.

Note

A Riemann problem is solved at each interface of the computational domain for the advection term. *Nektar++* provides ten different Riemann solvers, one exact and nine approximated. The exact one solves the problem using a Newton iterative method. The high accuracy of this method may imply a high computational cost. The approximated Riemann solvers do not take into account the full Riemann problem, these simplifications of the exact solver provide lower computational cost but lower accuracy.

**Task 2.2**

In the .xml file under the tag `SOLVERINFO`, define all the solver parameters as described above. These are declared as EQType, Projection, AdvectionType, DiffusionType, TimeIntegrationMethod, UpwindType, ViscosityType.

```

1 <SOLVERINFO>
2   <I PROPERTY="EQType" VALUE="NavierStokesCFE" />
3   <I PROPERTY="Projection" VALUE="DisContinuous" />
4   <I PROPERTY="AdvectionType" VALUE="WeakDG" />
5   <I PROPERTY="DiffusionType" VALUE="LDGNS" />
6   <I PROPERTY="TimeIntegrationMethod" VALUE="ClassicalRungeKutta4"/>
7   <I PROPERTY="UpwindType" VALUE="HLLC" />
8   <I PROPERTY="ProblemType" VALUE="General" />
9   <I PROPERTY="ViscosityType" VALUE="Constant" />
10 </SOLVERINFO>

```

Variables

In the `VARIABLES` tag we set the solution variable. For the 2D case we have:

```
1 <VARIABLES>
2   <V ID="0"> rho </V>
3   <V ID="1"> rhou </V>
4   <V ID="2"> rhov </V>
5   <V ID="3"> E </V>
6 </VARIABLES>
```

Note again the weak enforcement of the boundary conditions. The BCs are applied to the fluxes rather than to the non conservative variables of the problem. For further understanding, please check [A guide to the Implementation of the Boundary Conditions](#).

Boundary Conditions

The `BOUNDARYREGIONS` tag specifies the regions where to apply the boundary conditions.

```
1 <BOUNDARYREGIONS>
2   <B ID="0"> C[100] </B>
3   <B ID="1"> C[101] </B>
4   <B ID="2"> C[102] </B>
5   <B ID="3"> C[103] </B>
6 </BOUNDARYREGIONS>
```

The next tag is `BOUNDARYCONDITIONS` by which the boundary conditions are actually specified for each boundary ID specified in the `BOUNDARYREGIONS` tag. The boundary conditions have been set as explained in section 1.3

```
1 <!-- Wall -->
2   <REGION REF="0">
3     <D VAR="rho" USERDEFINEDTYPE="WallViscous" VALUE="0" />
4     <D VAR="rhou" USERDEFINEDTYPE="WallViscous" VALUE="0" />
5     <D VAR="rhov" USERDEFINEDTYPE="WallViscous" VALUE="0" />
6     <D VAR="E" USERDEFINEDTYPE="WallViscous" VALUE="0" />
7   </REGION>
8 <!-- Farfield -->
9   <REGION REF="1">
10    <D VAR="rho" VALUE="rhoInf" />
11    <D VAR="rhou" VALUE="rhoInf*uInf" />
12    <D VAR="rhov" VALUE="rhoInf*vInf" />
13    <D VAR="E" VALUE="pInf/(Gamma-1)+0.5*rhoInf*(uInf*uInf+vInf*vInf)" />
14  </REGION>
15 <!-- Inflow -->
16   <REGION REF="2">
17     <D VAR="rho" VALUE="rhoInf" />
18     <D VAR="rhou" VALUE="rhoInf*uInf" />
19     <D VAR="rhov" VALUE="rhoInf*vInf" />
20     <D VAR="E" VALUE="pInf/(Gamma-1)+0.5*rhoInf*(uInf*uInf+vInf*vInf)" />
21  </REGION>
22
23
24
```

```

25 <!-- Outflow -->
26 <REGION REF="3">
27   <D VAR="rho" VALUE="rhoInf" />
28   <D VAR="rhoU" VALUE="rhoInf*ulnf" />
29   <D VAR="rhoV" VALUE="rhoInf*vlnf" />
30   <D VAR="E" VALUE="pInf/(Gamma-1)+0.5*rhoInf*(ulnf*ulnf+vlnf*vlnf)" />
31 </REGION>

```

Note

As explained in section 2.3 Continuous Projection is not supported in the Compressible Flow Solver. Therefore, boundary conditions are specified through Dirichlet BCs and Neumann BCs are not supported.

The initial conditions have been set as explained in section 1.3.

```

1 <FUNCTION NAME="InitialConditions">
2   <E VAR="rho" VALUE="rhoInf"/>
3   <E VAR="rhoU" VALUE="rhoInf*ulnf" />
4   <E VAR="rhoV" VALUE="rhoInf*vlnf" />
5   <E VAR="E" VALUE="pInf/(Gamma-1)+0.5*rhoInf*(ulnf*ulnf+vlnf*vlnf)" />
6 </FUNCTION>

```

2.4 Artificial Viscosity

In order to stabilise the flow in the presence of flow discontinuities we utilise a shock capturing technique which makes use of artificial viscosity to damp oscillations in the solution, in conjunction with a discontinuity sensor to decide where the addition of artificial viscosity is needed.

Tip

In order to turn the NonSmooth artificial viscosity on:

- Include `ShockCaptureType` option in `SOLVERINFO` tag and set it to `NonSmooth`.
- Set the parameters `Skappa`, `Kappa` and `mu0` in the `PARAMETERS` tag. `mu0` is the maximum value for the viscosity, `Kappa` is half of the width of the transition interval and `SKappa` is value of the centre of the interval. The viscosity varies from 0 to the maximum values as the sensor goes from $Skappa-Kappa$ to $SKappa+Kappa$.
- The default values are: `Skappa` = -1.3; `kappa` = 0.2; `mu0` = 1.0.
- For further details, please read chapter 3 of [Mesh adaptation strategies for compressible flows using a high-order spectral/hp element discretisation](#)



Running the solver

The `CompressibleFlowSolver` can be run to solve the Cylinder Subsonic problem.



Task 3.1

Run the solver by typing the following command on the command line:
`$NEK/CompressibleFlowSolver CylinderSubsonic_NS.xml`



Tip

To reduce the solution time on computers with multiple processors, MPI can be used to run the simulation in parallel. Note that, for binaries compiled from source, the Cmake option `NEKTAR_USE_MPI` must have been set `ON`. To run in parallel, prefix the command in the previous task with `mpirun -np X`, replacing `X` by the number of parallel processes to use. For example, to use 32 processes:
`mpirun -np 32 $NEK/CompressibleFlowSolver CylinderSubsonic_NS.xml`

The simulation has now produced 10 `.chk` binary files and a final `.fld` binary file. These binary files contain the result of the simulation every 100 time-steps. This output interval has been chosen through the parameter `IO_CheckSteps` in `PARAMETERS` tag. Also, it is possible to note that every 100 time-steps the solver outputs the physical time of the simulation and the CPU time required for doing 100 time-steps. The interval of 100 time-steps is decided through the parameter `IO_InfoSteps`.



Tip

Stability plays a crucial role in the Compressible Flow solver. To ensure the solution is not polluted leading to numerical instabilities, for long simulations the `.chk` files can be checked before the simulation ends.

Simulation Results

Now that the simulation has been completed, we need to post-process the file in order to visualise the results. In order to do so, we can use the built-in post-processing routines within *Nektar++*. In particular we can use the following command:



Task 4.1

Convert the `.xml` and `.chk` files into a `.vtu` format by calling `$NEK/FieldConvert CylinderSubsonic_NS.xml CylinderSubsonic_NS.fld CylinderSubsonic_NS.vtu`.

Which generates a `.vtu` file that is a readable format for the open-source package Paraview. We can now open the `.vtu` file just generated and visualise it with Paraview. If we want to monitor the evolution of the simulation we can make an animation in Paraview by converting successive `.chk` files into `.vtu`



Task 4.2

Set the `FinTime` to 0.6 and run the simulation. In order to do that, define the number of steps *NumSteps* as the ratio of the *FinalTime* to the time-step *TimeStep* and set the *FinalTime*. Remember to use MPI in order to reduce the simulation time.

To create the animation we need to convert the `.xml` files into `.vtu` format. To avoid typing the same command several times, create a routine to create the different `.vtu` files. Once all the `.vtu` files are created (they are found in the completed folder), open them in paraview as a group (i.e File/Open and select all of them without expanding the tab).

If the final time is set to 0.6 and the `.chk` files are obtained every 400 steps. The animation created with the last 20 files should look like the `CylinderSubsonic_NS.ogv` video included in the completed folder.

Convert the .xml and .fld files into a .vtu format as shown in Task 4.1.

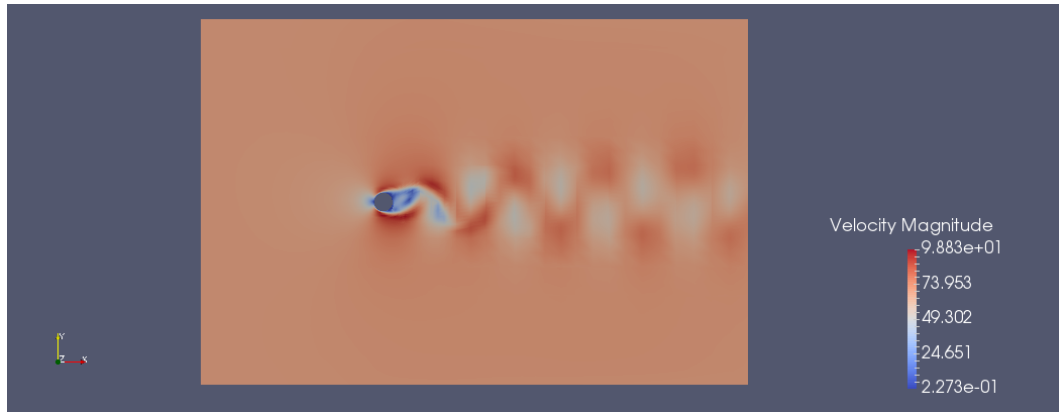


Figure 4.1 Instantaneous Velocity Flow Field

Calculate Vorticity

To perform the vorticity calculation and obtain an output data containing the vorticity solution, the user can run:



Task 4.3

Create a .fld file with the vorticity with the command:

```
$NEK/FieldConvert -m vorticity CylinderSubsonic_NS.xml  
CylinderSubsonic_NS.fld CylinderSubsonic_NS_vort.fld
```



Figure 4.2 Instantaneous Vorticity Flow Field

Extract Wall Shear Stress

To obtain the wall shear stress vector and magnitude, the user can run:

```
FieldConvert -m wss:bnd=0:addnormals=0 CylinderSubsonic_NS.xml
CylinderSubsonic_NS.fld CylinderSubsonic_NS_wss.fld
```

The option `bnd` specifies which boundary region to extract. In this case the boundary region ID of the cylinder is 0. If the `addnormals` is turned on, *Nektar++* additionally outputs the normal vector of the extracted boundary region.

In order to process the output file(s) you will need an .xml file of the same region. In order to do that we can use the NekMesh module extract:

```
NekMesh -m extract:surf=100 CylinderSubsonic_NS.xml bl.xml
```

Note, for NekMesh the surface ID we want to extract corresponds to the composite number of the cylinder surface -i.e 100.

To process the surface file one can use:

```
FieldConvert bl.xml CylinderSubsonic_NS_wss.fld
CylinderSubsonic_NS_wss.vtu
```

This command will generate a .dat file with the flow field information in the cylinder wall. It will produce the information of the density ρ , the fluxes ρu , ρv and E , the pressure p , the sound velocity a , the Mach number $Mach$, the sensor values $Sensor$, the shear values $Shear_x$, $Shear_y$ and $Shear_{mag}$ and the norms $norm_x$ and $norm_y$ for the different x and y coordinated along the cylinder. These files can be obtained from the completed folder.

This completes the tutorial.