UNIVERSITÀ DEGLI STUDI DELLA CALABRIA

Dipartimento di
Ingegneria Informatica, Modellistica, Elettronica e Sistemistica (DIMES)

Tesi di Dottorato di Ricerca
in Ambiente, salute, processi eco-sostenibili

Discontinuous Galerkin Methods for all speed flows

Ing. Salvatore Manuel Renda
Salvatore Manuel Renda

Supervisore	**Co-Supervisore**	**Coordinatore**
Ing. Carmine De Bartolo	Dr. Ralf Hartmann	Prof. Bruno de Cindio

Novembre 2013

DOTTORATO DI RICERCA XXVI Ciclo
SSD ING-IND/24 AMBIENTE, SALUTE E PROCESSI
ECO-SOSTENIBILI - LIFE SCIENCES

Discontinuous Galerkin Methods for all speed flows

by

Salvatore Manuel Renda.

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy.

Departement of Computer, Modelling,
Electronic and Information Systems Engineering,

November 2013.

Abstract

In this work we present a discontinuous Galerkin (DG) finite element method designed to improve the accuracy and efficiency of steady solutions of the compressible fully-coupled Reynolds Averaged Navier-Stokes (RANS) and $k-\omega$ turbulence model equations for all-speed flows using an implicit time integration method. The equations are solved with the backward Euler scheme, using the restarted generalized minimum residual (GMRES) method to approximately solve the linear system. The incomplete LU preconditioning is employed to accelerate the convergence of the linear solver.

The DG solution is extended to the incompressible limit by implementing a low Mach number preconditioning that affects both the time-derivative terms of the governing equations and the numerical dissipation of the Roe's Riemann solver, with the Harten's entropy fix, used to compute the numerical flux (full preconditioning technique). A new preconditioner based on a modified version of the Turkel preconditioning matrix, which takes into account the proposed formulation of the governing equations in conservative variables, is presented to improve the convergence process and accuracy.

At sonic speed the preconditioner reduces to the identity matrix thus recovering the non-preconditioned DG discretization and preserving the performance of the scheme for solving high-speed flows. An artificial viscosity term

is added to the DG discretized equations to stabilize the solution in the presence of shocks (transonic flows) when piecewise approximations of order of accuracy higher that one are used.

Various rescaling techniques are implemented to overcome ill-conditioning problems that, additionally to the low Mach number stiffness, can limit the performance of flow solvers. These approaches, through a proper manipulation of the governing equations, reduce unbalances between residuals due to the dependence on the size of elements in the computational mesh and due to an inherent difference between turbulent and mean-flow variables, influencing both the evolution of the CFL number and the robustness of the solver. Two kinds of rescalings are discussed, the *a priori* scaling, which affects both the time-step computation and the inexact solution of the linear systems, and the auto-scaling, which alters the linear systems only.

The performance of the method is demonstrated by solving inviscid, laminar and turbulent aerodynamic test cases. The computations are performed at different Mach numbers using various degrees of polynomial approximations to analyze the influence of the proposed numerical strategies on the accuracy, efficiency and robustness of a high-order DG solver at different flow regimes.

Contents

Chapter 1

Introduction

1.1 Motivation

Computer science and numerical simulation have became mandatory for the study of the complex physical phenomena, in engineering and in general in modern science. Nowadays, the application of numerical methods to fluid mechanics problems, the so-called Computational Fluid Dynamics (CFD), is routinely employed in the fields of aeronautics and aerospace, turbomachinery, transport and ship design. Furthermore, applications can be found even in various fields of numerical physics such as meteorology, oceanography, astrophysics, magnetohydrodynamics etc.

Aerodynamic computations performed by standard industrial codes are currently mainly based on the numerical solution of Reynolds Averaged Navier Stokes Equations (RANS) by means of formally second-order accurate finite volume (FV) schemes, due to their robustness and their favourable computational cost-accuracy ratio [1–5]. Let us first clarify that, mathematically, a numerical method is said to be of order k if the solution error e is proportional to the mesh size h to the power k, i.e., $e \propto h^k$ [6]. Although low-order schemes ($k \leq 2$) are widely used for engineering applications, the numerical accuracy provided is insufficient for turbulence, aeroacoustics, and many viscosity

dominant flows, such as boundary layer flows, vortical flows, shock-boundary layer interactions, heat flux transfers, etc. In particular, it has been shown that second-order discretization methods don't produce a satisfactory level of accuracy on meshes whose cell-size is typical for industrial applications. In the case of complex applications, in fact, very fine meshes with large number of grid points are required to obtain accurate solutions with second-order methods, leading to enormous computing times. Conversely, even on coarse meshes, high-order discretization methods allow to compute accurate solutions while significantly reducing the computational cost. Therefore, in recent years, several high-order methods have been emerging as handy tools to go beyond the standard accuracy of finite volume discretizations. In particular, in the United States the computational mathematics program of the Air Force Office of Scientific Research (AFOSR) and in Europe the project ADIGMA (Adaptive Higher-order Variational Methods for Aerodynamic Applications in Industry) [27] and IDIHOM [28] have devoted a significant research effort for the developement of high-order adaptive methods in academia as well as in industry. Many types of high-order methods have been developed in the CFD community to deal with a diverse range of problems [18–25]. Higher order ENO and WENO reconstructions on unstructured meshes have been implemented for standard finite volume methods with the main purpose of achieving high-order accurate solutions [7,8]. Nevertheless, these methods, based on extended stencils, are not used for industrial applications because of a certain instability of the algorithms. Therefore, research has been focused on new high-order schemes. One of the methods that received increasing attention in computational fluid dynamics in the last years because of many attractive features is the Discontinuous Galerkin (DG) finite element method [11]. The DG discretization approximates the numerical solution within the elements by piecewise continuous polynomial functions, whereas discontinuities of the solution are allowed at element interfaces where numerical flux functions are evaluated. Therefore, no global continuity is required in the computational

domain. In this respect, DG methods can be regarded as a hybrid of finite volume and classical finite element methods (FEM). Generally the lack of the global continuity constraint leads to more flexible discrete approximation. Thus DG schemes can be constructed for both structured and unstructured meshes allowing considerable flexibility in the use of non-conforming grids designed including pending or hanging nodes and in the use of elements with different order of accuracy, opening the way to a straightforward implementation of $hp-$refinement strategies. Due to their attractive capabilities, in the last decades the DG schemes are finding larger application in problems which traditionally were solved using the FV methods such as in various disciplines of numerical mathematics, physics and chemistry.

The availability of flow solvers for computing efficiently and accurately flows at all velocities is important for CFD in an industrial contex. Complex turbulent flows with simultaneous presence of high and low-speed regions are typical, for example, of automotive, turbomachinery and aircraft applications. Reliable simulations of such complex flow fields at a sufficient level of accuracy within low turn-around times require not only advances in discretization methods and physical modelling but also in numerical strategies in order to improve the performance of the solvers. One of the major reasons for the inefficiency of the numerical simulations is the stiffness that affects the system of equations: Euler, Navier-Stokes and RANS [29].

It's commonly known that stiffness in the equations arises for low Mach number and transonic flows [26]. The ill-conditioning of the discrete system is caused by a large disparity between acoustic and convective wave speeds which occurs as the Mach number approaches 0 or 1. As shown in [29, 30], standard algorithms for compressible flows, applied at the incompressible limit, suffer from slow convergence and in some cases from the lack of accuracy. To reduce the stiffness of the governing equations, some strategies should be applied in these cases [52–56]. In this work we focus our attention only on low

Mach number flows, even if the proposed approaches can be adapted also for transonic flows [26].

In addition to the low Mach and transonic stiffness, there are some other problems that can limit the performance of numerical schemes.

Firstly, an ill-conditioning of geometric nature, caused by a discrepancy of cell-sizes on the same mesh, generally affects the linear system, slowing down the convergence to steady state [31]. This is due to the scaling of the governing equations by the Jacobian of the coordinate transformation. As a result, the residual on cells where the mesh is coarser, is typically severals orders of magnitude higher than the residual on cells where the mesh is finer. Clearly, the scaling may have an influence on the linear solver and on the time step evolution, which is often based on the norm of residuals [34].

Secondly, in general the numerical solution of the RANS equations coupled with a turbulence model is challenging because of the extreme stiffness introduced by the turbulent transport equations. Indeed, mean flow variables typically do not exceed two, whereas turbulence model variables may be orders of magnitude higher. Thus, a large disparity occurs between the residual norm of the turbulence model equations and that of the mean-flow, leading to a further ill-conditioning that affects the time step computation and the efficiency of the linear solver [31, 34].

Lastly, it's worth to point out that high-order approximations of solutions with discontinuities, such as shocks, are oscillatory when discontinuities happen to be inside elements [44]. For this reason the discretization must include some technique for stabilization.

In this thesis we present a DG discretization of the 2D RANS and $k-\omega$ turbulence model equations to compute compressible all-speed flows. The conservative governing equations are written in terms of conservative variables and are iterated to steady state using an implicit scheme. The DG solution is extended to the incompressible limit by implementing a low Mach number

preconditioning that affects both, the time-derivative terms of the governing equations, through the action of a modified version of the Turkel preconditioning matrix [57], and the numerical dissipation of the Roe's Riemann solver, with the Harten's entropy fix [104], used to compute the numerical flux (full preconditioning technique). At sonic speed the preconditioner reduces to the identity matrix thus recovering the non-preconditioned DG discretization and preserving the accuracy and performance of the method for solving high-speed flows. An artificial viscosity term is added to the DG discretized equations to stabilize the solution in the presence of shocks (transonic flows) when piecewise polynomials of degree $p \geq 1$ are used. The shock capturing term is local and active in every element. However, the amount of artificial viscosity added to the scheme is almost negligible in smooth parts and large in elements containing the shock, such that large high-order elements, as typically used in high-order Discontinuous Galerkin methods, allow a sub-cell resolution of the shock. Finally, motivated by the results of Chisholm and Zingg [31], we extend the rescaling approaches to the high-order DG discretizations, trying to give a clearer understanding of their effect on the efficiency and robustness of DG flow solver.

The aim of this thesis is to contribute to the developement of a robust high-order accurate numerical scheme for compressible all-speed flows. A further objective is to improve the efficiency and the accuracy of the scheme for flow fields characterized by concurrent occurrence of compressible and incompressible effects. The implemented numerical techniques will always try to guarantee both the best numerical efficiency and best accuracy as possible. This will be performed depending on the local Mach number and on the governing equations.

Please refer to section 1.2 "Background" for a comprehensive review and a more detailed discussion about the DG method and all the numerical strategies adopted in this work.

Remark: We note that the numerical methods described in this thesis have been implemented and tested based on two separate Discontinuous Galerkin flow solvers: *(i)* the `PADGE` code of Hartmann *et al.* (DLR) [59] which is based on the `deal.II` library [61] and *(ii)* the `MIGALE` code of Bassi *et al.* (University of Bergamo) [62]. In particular, the low Mach number preconditioning has been implemented in both, the `PADGE` and the `MIGALE` code, whereas rescaling techniques have been developed in the `PADGE` code only. Therefore, we remark that most of the numerical results which are presented in this thesis have been obtained using the `PADGE` code.

1.2 Background

1.2.1 High-Order Methods

In the past 20 to 30 years, high-order methods have gained great attention in the CFD community because of their potential in providing higher accuracy with lower cost than low-order methods. A numerical method is called high order, when its order of accuracy is at least greater than two. The potentialities of high-order schemes are due to the fact that, to predict numerical solutions, a high-order method may be more efficient than a low-order one because it can achieve the prescribed error threshold on a much coarser mesh. Despite second-order methods have been the workhorse for CFD, many flow problems are still too expensive or out of their reach. On the other hand, high-order methods have been shown to predict situations well containing both discontinuities and rich structures in the smooth part of the solutions, such as the Rayleigh-Taylor instability simulation, shock interaction with vortices, and direct simulation of compressible turbulence. The most popular high-order methods are the spectral ones, whose basis functions are chosen as sums of sinusoids [16]. Although spectral methods are computationally less expensive than finite element methods, they become less accurate for problems with

complex geometries and discontinuous solutions [17]. Motivated by the potentialities emerged from these high-order schemes and with the purpose to overcome their limitations, in the early 1980s the research moved towards the so called p-type finite element method. The main feature of p-type FEM is that, for a given grid spacing h, the polynomial degree p is increased to decrease the error. One of the results of the studies that Babuska *et al.* [37] carried out in the 1981, applying this method to elasticity problems, was that based on degrees of freedom, the rate of convergence of the p-type method cannot be slower than that of the h-type and that, in the case of a singularity problem, the rate of convergence of the p-type is twice as fast. Starting from these first studies significant research effort has been aimed at developing high-order accurate methods. For a general review of these methods, refer to [38–41].

1.2.2 Discontinuous Galerkin Methods

Among high-order accurate methods, one of those that has become most popular in the last years is the Discontinuous Galerkin (DG) finite element method. Like continuous finite element methods, the DG methods can increase the accuracy of the numerical solution simply by increasing the degree of the polynomial approximation whithin each element, whereas, as in finite volume methods, at the element interfaces upwind discretizations of fluxes are employed. This kind of discretization enforces no global continuity, allowing the treatment of each element as a separate entity that communicates with the adjacent elements only through the numerical fluxes, giving a compact space discretization. This feature of the scheme is particularly advantageous when an implicit time integration scheme is employed and/or for a parallel implementation of the method. Moreover, the compact formulation of the scheme can be applied close to the boundary, without any special treatment, increasing the robustness and the accuracy of boundary conditions. All these aspects are crucial in

order to deal with shock waves, turbulence modelling and complex geometries, typical of industrial applications.

The Discontinuous Galerkin Method was originally introduced by Reed and Hill [42] in 1973 for neutron transport problems. The method's accuracy and stability properties have been rigorously proven by Johnson and Pitkarata [9], Cockburn and Shu [24], Cockburn *et al.* [10], Cockburn *et al.* [10] and Jiang and Shu [13] for arbitrary element shapes, any number of spatial dimensions, and even for non-linear problems. With respect to compressible and incompressible flows, implicit and explicit high order DG solvers of the Euler and Navier-Stokes equations have been applied to both, model problems and complex 3D applications, and are now rather well-established. An implicit DG method for the coupled RANS (Reynolds-Averaged Navier-Stokes) and $k-\omega$ turbulence model equations was developed by e.g. Bassi *et al.* [62,81] and Hartmann [45].

1.2.3 Turbulence modelling

Turbulent flows are very common in industrial applications. Some examples include oil transport in pipelines, flows through pumps and turbines, flows in boat wakes and around aircrafts.

In contrast to laminar flow, the main characteristic of a turbulent flow is that the molecules move in a chaotic way along complex irregular paths. As consequence, this strong chaotic motion causes an intense mixing of the various layers of the fluid. Moreover, turbulent flows lead to higher skin friction and heat transfer as compared to laminar ones due to an increased momentum and energy exchange between the molecules and solid walls [33].

Despite the continuous increase of computational resources, a direct numerical simulation (DNS) of Navier-Stokes equations that solves all the turbulent scales can be carried out only for simple geometries and at low Reynolds number. As a result, turbulent phenomena need to be modelled and over the years a large variety of turbulence models has been developed.

Alternative approaches to DNS technique are Large Eddy Simulations (LES) [73] and Detached Eddy Simulations (DES) [74]. Nevertheless, these models, in which large turbulent scales are resolved and the small scales are modelled, turn out to be highly time consuming when the analysis has to be extended to industrial applications, because of the more complex nature of both the flow and the geometry. For these reasons, three-dimensional turbulence models for flows of industrial interest are usually based on the Reynolds Averaged Navier-Stokes equations (RANS), due to their higher robustness and lower computational cost. In the RANS, all turbulent scales are modelled, with a considerable reduction in the computational effort but, also, in the numerical accuracy. The shortcomings of RANS models have been broadly discussed in the literature [68–72] and their numerical validation is necessary especially for industrial applications.

Actually, the most common RANS models in industrial CFD are $k-\varepsilon$ [76] and Wilcox's $k-\omega$ [77, 78], both based on the solution of a turbulent kinetic energy transport equation in combination with another model equation, the dissipation rate and the specific dissipation rate transport equation, respectively. A blend of these two approaches has been implemented in the $k-\omega$ Shear Stress Transport (SST) model of Menter [79], which tries to combine the positive features of both the previous two, employing the $k-\omega$ approach in the sublayer and logarithmic part of the boundary layer and the $k-\varepsilon$ one in the remaining part of it. Finally, the recent growth of computational power allowed the developement of more complex turbulent models. In Reynolds Stress Models (RSM), transport equations are solved for the individual Reynolds stress components. As a consequence, the turbulent viscosity hypothesis, one of the major defects of $k-\varepsilon$ and $k-\omega$ models, is eliminated [85, 86].

In this work, we use the Wilcox's $k-\omega$ model implemented in the framework of DG schemes by Bassi *et al.* [81]. The RANS and $k-\omega$ equations are supplemented with some form of limiting of the magnitude of the computed turbulent quantities in order to prevent the blow-up of the simulations, which

turned out to be a particularly important issue for higher order computations when solution is advanced in time by means of implicit integration schemes.

1.2.4 Preconditioning Techniques

Algorithms used for compressible flows suffer from a lack of accuracy and slow convergence when solving low Mach number flows in which the density is almost constant [29]. The reason for the bad convergence is the large disparity between acoustic and convective wave speeds that causes the governing equations to be ill-conditioned. The decreasing accuracy results from a lack of artificial dissipation for small Mach numbers, as addressed for upwind schemes by Guillard and Viozat [30] with their asymptotic analysis of the Euler equations. The most general approach to overcome the stiffness problem is based on the preconditioning strategy. This technique artificially modifies the acoustic wave speeds of the governing equations reducing the condition number and improving the convergence process. However, the time derivative preconditioning destroys the time accuracy and can be applied to steady-state simulations only. To overcome this limitation, dual time-stepping technique may be employed [48]. Furthermore, also the accuracy of upwind schemes can be improved by preconditioning, modifying the inviscid dissipation term of the numerical flux function. Numerous studies have been carried out on these topics in the past. In particular, in the context of high-order methods it was shown in [52] that, because the explicit time integration schemes suffer from severe time stepping restrictions computing low speed flows, an implicit time integration method is more appropriate for low Mach number computations. Moreover, Bassi *et al.* found [89] that preconditioning only needs to be applied to the numerical flux function (flux preconditioning technique) for guaranteeing accuracy and convergence rate improvements. Some of the most recognized local preconditioners for laminar flows were proposed by Choi and Merkle [49], Turkel [52, 57], Lee and van Leer [26] and Weiss and Smith [48].

Extensions to turbulent flows with two equations turbulence models have been proposed by Colin [90] and Yoo [91].

In this work, we apply the full preconditioning approach, that alters both the time-derivative terms of the governing equations and the numerical flux function, to the fully coupled RANS $k - \omega$ equations, in order to accurately and efficiently compute steady low Mach number flows.

1.2.5 Rescaling techniques

A further issue which can greatly affect the performance of flow solvers concerns the disparity in the magnitude of entries in the residual vector as well as of entries in the Jacobian matrix, which arises because of the differences between both the sizes of elements in the grid and the magnitude of mean-flow and turbulent variables.

Addressing scaling problems might have a significant effect on the performance of inexact Newton methods. Indeed, a prescribed reduction tolerance in the residual of the linear system could be achieved by reducing only the residual of the larger grid elements and/or the residual of the turbulence model transport equations, while the residuals of both the smaller elements, typically clustered near the wall, and the mean-flow equations, which are orders of magnitude smaller than the previous ones, might increase leading to non-convergence of the Newton iterations. Moreover, a poor scaling of variables and equations can be detrimental in the evolution of the time step, because the evolution of the CFL number is often based on the norm of the residual, e.g. SER techniques [34].

A numerical strategy to cope with these problems is based on rescaling techniques, as proposed by Chisholm and Zingg in [31]. In particular, there are two kinds of approaches: inherent scaling and auto-scaling. The former is used to fix *ab initio* scaling problems, such as the geometric scaling and the inherent scaling differences between turbulent and mean-flow variables. The latter approach rescales the linear problem of each non-linear iteration to address discrepancies between the residual norms of the various equations. The rescaling strategies are expected to lead to some improvements in convergence and robustness of the solver.

1.2.6 Shock capturing

The numerical dissipation introduced by the DG discretization is not sufficient to stabilize the solution in the presence of shocks when piecewise higher than first-order approximations are used; indeed for higher order schemes a further explicit dissipation must be added to obtain stable solutions. Several strategies inspired by finite volume schemes have been proposed in order to accurately represent shocks with high-order methods. For instance, a straightforward approach is based on reducing the polynomial degree in those elements flagged by a sensor which suggests cells lying in the shock region. Here, the amount of numerical dissipation added by the scheme is increased, therefore an adaptive mesh refinement close to the shock only may alleviate the problem of lack of accuracy caused by lower order elements. Several other approaches to control the oscillations are discussed in [7, 8, 87, 88].

Anyway, one of the most used strategies, which proved to be effective in the framework of DG methods, consists of adding to the DG discretized equations an artificial viscosity term that aims at controlling the high-order modes of the numerical solution within elements while preserving as much as possible the spatial resolution of the discontinuities. In this approach, the computed shock capturing term is local and active in every element. In particular the amount of artificial viscosity added to the scheme is almost negligible in smooth parts and large in non-smooth parts. Unlike in finite volume schemes, where the shock is spread over several elements in the mesh, in DG schemes the artificial viscosity method has the capabilities to resolve the shock typically in only one cell. Some shock capturing schemes have been presented by Hartmann [45] and Bassi [44] in the context of DG schemes for inviscid and turbulent compressible flows.

1.3 Outline of Thesis

This thesis deals with a high-order accurate discontinuous finite element method for the numerical solution of the compressible Reynolds-averaged Navier-Stokes (RANS) and $k-\omega$ turbulence equations. Low Mach number preconditioning, shock-capturing and rescaling techniques are discussed and implemented with the aim of increasing efficiency, robustness and accuracy of a DG solver for a large range of flow conditions.

The outline of the present thesis is as follows:

- In Chapter 2 we present the physical model adopted in this work: the RANS and $k-\omega$ turbulence equations in conservative form.
- In Chapter 3 we describe the Discontinuous Galerkin discretization of the governing equations and we give details related to implicit time integration.
- In Chapter 4 we present the low Mach number preconditioning for steady state computations of laminar and turbulent flows.
- In Chapter 5 we present the rescaling techniques. In particular, we consider two kinds of approaches: the inherent scaling and the auto-scaling.

Finally, at the end of this thesis the main conclusions of the study are presented in a short "Conclusions" section.

Chapter 2

Physical model

In this chapter we describe the governing equations for two-dimensional low and high speed flows. We consider the fully coupled Reynolds-Averaged Navier-Stokes and $k-\omega$ turbulence model equations written in conservative form. The equations governing laminar flows can be obtained from the RANS model switching off the $k-\omega$ transport equations and forcing the turbulent viscosity to be zero. Finally, some considerations concerning the non-dimensionalization of the equations are given.

2.1 The compressible Reynolds-Averaged Navier-Stokes and $k-\omega$ turbulence equations

The compressible Reynolds-averaged Navier-Stokes (RANS) and $k-\omega$ turbulence model equations can be written in conservative form on the domain $\Omega \subset R^2$, as

$$\mathbf{\Gamma}\frac{\partial \mathbf{u}}{\partial t} + \boldsymbol{\nabla}\cdot\mathbf{F}_c - \boldsymbol{\nabla}\cdot\mathbf{F}_v - \boldsymbol{\nabla}\cdot\mathbf{S} = 0, in\ \ \Omega \tag{2.1}$$

with the vector of conservative variables $\mathbf{u} = \{\rho, \rho u_j, \rho E, \rho k, \rho\widetilde{\omega}\}^T$, where ρ denotes the density, $\mathbf{v} = \{u_1, u_2\}^T$ the velocity vector, E the total energy, k the turbulent kinetic energy and ω the specific value of turbulence dissipation. According to [81], the model employs the variable $\tilde{\omega} = ln\,(\omega)$ instead of ω to obtain a smoother near-wall distribution and to guarantee the positivity of ω.

The matrix $\mathbf{\Gamma}$ is set equal to the identity matrix $\mathbf{I}$ for the system of equations written in conservative variables $\mathbf{u}$. However, the preconditioned RANS $k-\omega$ equations can be obtained from Eq. (2.1) setting $\mathbf{\Gamma}$=$\mathbf{P}_u$, where $\mathbf{P}_u$ denotes the preconditioning matrix in conservative variables. Otherwise, it is possible to use the set of primitive variables $\mathbf{q} = \{p, u_j, T, k, \widetilde{\omega}\}^T$, by replacing in Eq. (2.1) $\mathbf{u}$ with $\mathbf{q}$ and $\mathbf{\Gamma}$ with $\frac{\partial \mathbf{u}}{\partial \mathbf{q}}$, where $\frac{\partial \mathbf{u}}{\partial \mathbf{q}}$ is the transformation matrix from conservative to primitive variables. Finally, the preconditioned formulation of the system Eq. (2.1) rewritten in primitive variables $\mathbf{q}$ is obtained for $\mathbf{\Gamma} = \frac{\partial \mathbf{u}}{\partial \mathbf{q}}\mathbf{P}_q$, where $\mathbf{P}_q$ denotes the preconditioning matrix in primitive variables [102]. Please, refer to Chapter 4 for a detailed description of preconditioners. The convective fluxes $\mathbf{F}_c = (\mathbf{f}_c^1, \mathbf{f}_c^2)$ and the viscous ones $\mathbf{F}_v = (\mathbf{f}_v^1, \mathbf{f}_v^2)$ are given by

$$\mathbf{f}_c^j = \begin{pmatrix} \rho u_j \\ \rho u_j u_i + \left(p + \frac{2}{3}\rho k\right)\delta_{ij} \\ \left(\rho H + \frac{2}{3}\rho k\right) u_j \\ \rho u_j k \\ \rho u_j \widetilde{\omega} \end{pmatrix}, \quad \mathbf{f}_v^j = \begin{pmatrix} 0 \\ \hat{\tau}_{ij} \\ u_i\hat{\tau}_{ji} + \mathcal{K}\frac{\partial T}{\partial x_j} + (\mu + \sigma_k\overline{\mu}_t)\frac{\partial k}{\partial x_j} \\ (\mu + \sigma_k\overline{\mu}_t)\frac{\partial k}{\partial x_j} \\ (\mu + \sigma_\omega\overline{\mu}_t)\frac{\partial \widetilde{\omega}}{\partial x_j} \end{pmatrix},$$

$$j = 1, 2.$$

Here H is the total enthalpy given by $H = E + \frac{p}{\rho}$ and the pressure p can be computed using the equation of state of an ideal gas,

$$p = (\gamma - 1)\,\rho\left(E - \frac{\mathbf{v}^2}{2} - k\right),$$

where $\gamma = \frac{c_p}{c_v}$ denotes the ratio of the specific heat capacities at constant pressure c_p and at constant volume c_v and equals to 1.4 for dry air. In the heat conduction term the termal conductivity $\mathcal{K}$ is given by $\mathcal{K} = c_p\left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)$ and the temperature T by $c_v T = E - \frac{1}{2}\mathbf{v}^2 - k$. Furthermore, $Pr = 0.72$ and $Pr_t = 0.9$ are the molecular and turbulent Prandtl numbers and μ and μ_t are the molecular and turbulent viscosities, respectively. The sum of pressure p and $\frac{2}{3}\rho k\delta_{ij}$ is called effective pressure $p_{eff} = p + \frac{2}{3}\rho k\delta_{ij}$.

The total stress tensor, determined by the sum of the viscous stress tensor μS_{ij} and the Reynolds stress tensor $\tau_{ij} = \mu_t S_{ij} - \frac{2}{3}\rho k\delta_{ij}$, is defined as

$$\overline{\tau}_{ij} = (\mu + \mu_t)\,S_{ij} - \frac{2}{3}\rho k\delta_{ij}, \qquad i, j = 1, 2,$$

where S_{ij} is given by

$$S_{ij} = \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}, \qquad i, j = 1, 2,$$

We note that because of the contribution $\frac{2}{3}\rho k\delta_{ij}$ of the Reynolds stress tensor to the convective part, $\hat{\tau}_{ij} = \overline{\tau}_{ij} + \frac{2}{3}\rho k\delta_{ij}$ is used to compute the viscous fluxes. Finally, the source term $\mathbf{S}\left(\mathbf{u}, \boldsymbol{\nabla}\mathbf{u}\right)$ is given by

$$\mathbf{S} = \begin{pmatrix} 0 \\ \mathbf{0} \\ 0 \\ -\beta_k \rho \overline{k} e^{\widetilde{\omega}_r} + \tau_{ij}\frac{\partial u_i}{\partial x_j} \\ \alpha_\omega \tau_{ij}\frac{\omega}{k}\frac{\partial u_i}{\partial x_j} - \beta_\omega \rho e^{\widetilde{\omega}_r} + \left(\mu + \sigma_\omega \overline{\mu}_t\right)\frac{\partial \widetilde{\omega}}{\partial x_k}\frac{\partial \widetilde{\omega}}{\partial x_k} \end{pmatrix},$$

where the turbulent eddy viscosity μ_t can be calculated as $\mu_t = \frac{C_\mu \rho k}{\omega}$ and $\alpha_\omega = \frac{5}{9}$, $\beta_k = \frac{9}{100}$, $\beta_\omega = \frac{3}{40}$, $\sigma_k = \frac{1}{2}$, and $\sigma_\omega = \frac{1}{2}$ stand for the closure coefficients of the high-Reynolds number Wilcox $k - \omega$ model [77, 78].

In order to prevent the possibility of breakdown during the computations, in addition to the use of the logarithm of the turbulence variable ω introduced in [84], further realizability conditions are addressed for the turbulent stresses. In particular, the variables k and $\tilde{\omega}$ are limited from below by

$$\overline{k} = \max(k, 0), \qquad \tilde{\omega_r} = \max\{\tilde{\omega}, \tilde{\omega_{r0}}\},$$

where $\tilde{\omega_{r0}}$ defines the lower bound on $\widetilde{\omega}$ that ensures the positivity of normal turbulent stresses and the fulfillment of the following inequality, similar to that proposed in [44, 81]

$$e^{\tilde{\omega_{r0}}} - \frac{3}{2}C_\mu S_{ii} \geq 0, \quad i = 1, 2,$$
$$\left(e^{\tilde{\omega_{r0}}}\right)^2 - \frac{3}{2}C_\mu \left(S_{ii} + S_{jj}\right) e^{\tilde{\omega_{r0}}} + \frac{9}{4}{C_\mu}^2 \left(S_{ii}S_{jj} - {S_{ij}}^2\right) \geq 0, \;\; i, j = 1, 2.\, i \neq j.$$

The limited turbulent eddy viscosity $\overline{\mu}_t$ is computed by the following expression

$$\overline{\mu}_t = \alpha^* \rho \overline{k} e^{-\widetilde{\omega}_r}.$$

2.2 Non-dimensionalization

In this work we consider the governing equations in non-dimensionalized variables based on the freestream (reference) density $\rho_\infty = 1$, pressure $p_\infty = 1$ and temperature $T_\infty = 1$. Therefore, the specific heat capacities at constant volume c_v, constant pressure c_p and the molecular viscosity μ are given by

$$c_v = \frac{1}{\gamma - 1}, \qquad c_p = \frac{\gamma}{\gamma - 1},$$
$$\mu = \frac{\sqrt{\gamma} M_\infty}{Re_\infty},$$

where M_∞ and Re_∞ are the Mach and Reynolds numbers at the freestream, respectively. Reference values for the other quantities are derived from these by functional relationships. With this choice of non-dimensionalized variables, all the governing equations remain unchanged, except that the variables are now understood to be non-dimensionalized.

Chapter 3

Discontinuous Galerkin Formulation

In this chapter we present a high-order accurate DG discretization of the Reynolds-Averaged Navier-Stokes (RANS) and the $k-\omega$ turbulence model equations. Following the method of lines approach, we employ a separate discretization, first in space and then in time. The Roe's approximate Riemann solver with the Harten's entropy fix is introduced for computing the numerical fluxes at element interfaces. The system of equation is iterated to steady state by means of the first-order accurate backward Euler scheme.

3.1 DG discretization of the RANS and $k-\omega$ equations

The weak form of the RANS $k-\omega$ equations is obtained multiplying Eq. (2.1) by a test function $\mathbf{v}$ and integrating by parts

$$\int_{\Omega} \mathbf{v}^T \boldsymbol{\Gamma} \frac{\partial \mathbf{u}}{\partial t} d\mathbf{x} - \int_{\Omega} \boldsymbol{\nabla} \mathbf{v}^T \cdot \mathbf{F}\left(\mathbf{u}, \boldsymbol{\nabla}\mathbf{u}\right) d\mathbf{x} + \int_{\partial\Omega} \mathbf{v}^T \mathbf{F}\left(\mathbf{u}, \boldsymbol{\nabla}\mathbf{u}\right) \cdot \mathbf{n} \, d\sigma - \int_{\Omega} \mathbf{v}^T \mathbf{S}\left(\mathbf{u}, \boldsymbol{\nabla}\mathbf{u}\right) d\mathbf{x} = \mathbf{0}, \qquad \forall \mathbf{v} \in H\left(\Omega\right)^1, \tag{3.1}$$

where Ω is the domain with boundary $\partial\Omega$, $\mathbf{v}$ denotes a vector of any arbitrary, sufficiently smooth, test functions and $\mathbf{F}$ is the algebraic sum of inviscid and viscous fluxes, $\mathbf{F} = \mathbf{F}_c - \mathbf{F}_v$.

For the sake of presenting the DG space discretization of the RANS and turbulence model equations, we define ${\mathbf{V}_h}^n$ to be the space of discontinuous vector-valued polynomials of degree n, on a triangulation $\tau_h = \{K\}$ of an approximation Ω_h of the domain Ω into non-overlapping elements such that $\Omega_h = \bigcup_{K\in\tau_h} K$. Thus, the solution and test functions space is defined by

$${\mathbf{V}_h}^n = \{\mathbf{v}_h \in L^2\left(\Omega_h\right) : \mathbf{v}_h|_K \in P_n\left(K\right) \forall K \in \tau_h\},$$

where $P_n\left(K\right)$ is the space of polynomial functions of degree at most n.

The DG approximation is obtained by evaluating the viscous flux, $\mathbf{F}_v$, according to the BR2 scheme [65, 81], as

$$\mathbf{F}_v|_{\Omega_h} = \mathbf{F}_v\left(\mathbf{u}_h, \boldsymbol{\nabla}\mathbf{u}_h + \boldsymbol{\mathcal{R}}\left(\left[\!\left[\mathbf{u}_h\right]\!\right]_0\right)\right),$$

$$\mathbf{F}_v|_{\Gamma_h} = \mathbf{F}_v\left(\mathbf{u}_h, \boldsymbol{\nabla}\mathbf{u}_h + \eta_e \boldsymbol{\mathcal{R}}_e\left(\left[\!\left[\mathbf{u}_h\right]\!\right]\right)\right),$$

where Γ_h denotes the union of the set of internal, Γ_h^0, and boundary, $\partial\Omega_h$, element edges, E, such that $\Gamma_h = \Gamma_h^0 \bigcup \partial\Omega_h$. Then the discrete problem takes the following form: find $\mathbf{u}_h \in {\mathbf{V}_h}^n$ such that

$$\int_{\Omega_h} \mathbf{v}_h^T \boldsymbol{\Gamma} \frac{\partial \mathbf{u}_h}{\partial t} d\mathbf{x} - \int_{\Omega_h} \boldsymbol{\nabla} \mathbf{v}_h^T \cdot \left(\mathbf{F}_c\left(\mathbf{u}_h\right) - \mathbf{F}_v\left(\mathbf{u}_h, \boldsymbol{\nabla}\mathbf{u}_h + \boldsymbol{\mathcal{R}}\left(\left[\!\left[\mathbf{u}_h\right]\!\right]_0\right)\right)\right) d\mathbf{x}$$

$$+\int_{\Gamma_h}[\![\mathbf{v}_h]\!]\cdot\left[\hat{\mathbf{f}}_c\left({\mathbf{u}_h}^{\pm},\mathbf{n}\right)-\mathbf{F}_v\left(\mathbf{u}_h,\boldsymbol{\nabla}\mathbf{u}_h+\eta_e\boldsymbol{\mathcal{R}}_e\left([\![\mathbf{u}_h]\!]\right)\right)\right]d\sigma$$

$$-\int_{\Omega_h}\mathbf{v}_h^T\mathbf{S}\left(\mathbf{u}_h,\boldsymbol{\nabla}\mathbf{u}_h+\boldsymbol{\mathcal{R}}\left([\![\mathbf{u}_h]\!]_0\right)\right)d\mathbf{x}=0,$$

$$\forall\mathbf{v}_h\in\mathbf{V}_h. \tag{3.2}$$

Here, $()^+$ and $()^-$ denote the values of any quantity evaluated from inside and outside faces of an element K, and $\mathbf{n}$ is the unit outward normal vector to K, see Figure 3.1.

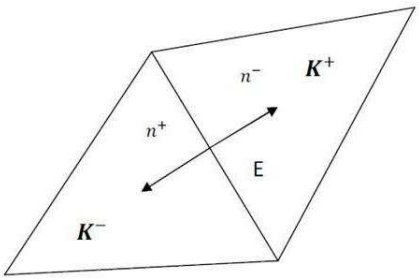


Figure 3.1: Two elements K^+ and K^- sharing edge E

$\boldsymbol{\mathcal{R}}_e\left([\![\mathbf{u}_h]\!]\right)$ and $\boldsymbol{\mathcal{R}}\left([\![\mathbf{u}_h]\!]_0\right)$ are the local and global lifting operators, respectively, given by

$$\int_{\Omega_h}\mathbf{v}_h^T\cdot\boldsymbol{\mathcal{R}}_e\left([\![\mathbf{u}_h]\!]\right)d\mathbf{x}=-\int_{\mathrm{E}}\left\{\mathbf{v}_h^T\right\}\cdot[\![\mathbf{u}_h]\!]_0 d\sigma,$$

$$\boldsymbol{\mathcal{R}}\left([\![\mathbf{u}_h]\!]_0\right)=\sum_{\mathrm{E}}\boldsymbol{\mathcal{R}}_e\left([\![\mathbf{u}_h]\!]\right),$$

with jump $[\![\cdot]\!]$ and average $\{(\cdot)\}$ trace operators, acting componentwise when applied to a vector, defined as

$$[\![\mathbf{u}_h]\!]\triangleq\left(\mathbf{u}_h\mathbf{n}\right)^+-\left(\mathbf{u}_h\mathbf{n}\right)^-,\qquad\{\mathbf{u}_h\}\triangleq\frac{\left(\mathbf{u}_h\right)^+-\left(\mathbf{u}_h\right)^-}{2}.$$

Each local lifting operator is not null only on one or two elements K^+ and K^- (respectively right and left state) adjacent to the generic edge E. η_e is called penalty parameter and its lower bound is established as the number of neighbours of the generic element K in order to guarantee the stability of

the method. Nevertheless, according to [75] lower values of η_e can provide better convergences without spoiling the accuracy of the numerical solution, due to an improved condition number of the linear system matrix. The BR2 viscous flux discretization is compact because each element K only couples the nearest neighbor elements, simplifying an implicit implementation of the method. Finally, in the present work the approximation of the convective interface flux, $\hat{\mathbf{f}}_c\left(\mathbf{u}_h{}^{\pm}, \mathbf{n}\right)$, is performed using the Roe's scheme [3, 4] with the Harten's entropy fix [104, 105].

3.1.1 Oscillation control for flows with shock

Spurious oscillation can be generated by the DG method in the vicinity of discontinuities of the solution, when polynomials of higher degree ($p > 0$) are used. To suppress oscillatory behaviour and computing accurate numerical solutions, some sort of non-linear numerical dissipation, which does not adversely affect the standard accuracy of the scheme, is needed. In particular, we augment the DG discretization Eq. (3.2) with a stabilization term based on artificial viscosity [45], generally given by

$$\int_{\Omega_h} \varepsilon_h\left(\mathbf{u}_h\right) \boldsymbol{\nabla} \mathbf{v}_h \cdot \boldsymbol{\nabla} \mathbf{u}_h d\mathbf{x}. \tag{3.3}$$

with $\varepsilon_h\left(\mathbf{u}_h\right)$ called artificial viscosity. The artificial viscosity entries can be computed, for each element, as

$$\varepsilon_{klm}|_K = \begin{cases} C_\varepsilon \delta_{kl} \widetilde{h}_k^2 f_p\left(\mathbf{u}_h\right) \left|R_p\left(\mathbf{u}_h\right)\right| & m = 0, ..., 3 \\ 0 & m = 4, 5 \end{cases} \qquad k, l = 1, 2. \tag{3.4}$$

Here the scalar residual term $R_p\left(\mathbf{u}_h\right)$ is given by

$$R_p\left(\mathbf{u}_h\right) = \sum_{m=1}^{d} \frac{\partial p}{\partial u_m} \frac{R_m\left(\mathbf{u}_h\right)}{p},$$

where $R_m\left(\mathbf{u}_h\right)$ denotes the residual of cell for the component m and $\frac{\partial p}{\partial u_m}$ is the partial derivative of the pressure with respect to the conservative variable

u_m. Moreover, $\widetilde{h}_k$ is the directional element size given by

$$\widetilde{h}_k = \frac{h_k}{n+1},$$

where h_k denotes the dimension of the element K in the x_k-coordinate direction and n polynomial degree. The pressure sensor $f_p(\mathbf{u}_h)$ defined as

$$f_p(\mathbf{u}_h) = \frac{h_{cell}|\boldsymbol{\nabla} p|}{p + \varepsilon'},$$

with

$$h_{cell} = (h_1 h_2)^{\frac{1}{2}}, \qquad \varepsilon' = 10^{-12},$$

is small in smooth parts and large in non-smooth parts of the flow solution, and it decreases the effect of the artificial viscosity in smooth parts with steep gradients. Finally C_ε is a tunable parameter of the artificial viscosity [45].

3.2 Numerical flux function

3.2.1 Roe's approximate Riemann solver

The convective flux at the element interfaces can be computed between the left and right state of the discontinuous discrete solution by solving a local Riemann problem. In order to reduce the computational effort for the exact solution of the Riemann problem (Godunov scheme), Riemann approximate solvers can be used. In this work we employ the Roe's scheme, which is based on a linearization of the Riemann problem. The Roe's method is often applied because of its high accuracy for both boundary layers and shock resolution. The Roe flux is given by

$$\mathbf{H}\left(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n}\right) = \frac{1}{2}\left(\mathbf{F}_c(\mathbf{u}^+)\cdot\mathbf{n} + \mathbf{F}_c(\mathbf{u}^-)\cdot\mathbf{n} - \mathbf{A}\left(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n}\right)\Delta\mathbf{u}\right), \qquad (3.5)$$

and can be regarded as an average between the convective fluxes $\mathbf{F}_c(\mathbf{u}^+)$ and $\mathbf{F}_c(\mathbf{u}^-)$ computed at the interior and exterior side of the face, plus a dissipation term $\mathbf{A}(\mathbf{u}^+, \mathbf{u}^-, \mathbf{n})\Delta\mathbf{u}$ with $\Delta\mathbf{u} = \mathbf{u}^- - \mathbf{u}^+$, to stabilize the scheme. The

matrix $\mathbf{A}$, called Roe matrix or dissipation matrix, is computed using the so-called Roe-averaged variables as

$$\mathbf{A}\left(\mathbf{u}^{+},\mathbf{u}^{-},\mathbf{n}\right)=\left|\left(\frac{\partial\mathbf{F}_{c}}{\partial\mathbf{u}}\cdot\mathbf{n}\right)\right|=(\ \mathbf{T}|\mathbf{\Lambda}|\mathbf{T}^{-1}),$$

where $\mathbf{\Lambda}=\mathrm{diag}\{\alpha_i\}$ in the original scheme is set equal to the matrix of eigenvalues $\mathrm{diag}\{\lambda_i\}$ with $\lambda_i=\{\mathbf{v}\cdot\mathbf{n},\mathbf{v}\cdot\mathbf{n},\mathbf{v}\cdot\mathbf{n},\mathbf{v}\cdot\mathbf{n},\mathbf{v}\cdot\mathbf{n}+a_t,\mathbf{v}\cdot\mathbf{n}-a_t\}$ [105]. The eigenvalues of $\left(\frac{\partial\mathbf{F}_c}{\partial\mathbf{u}}\cdot\mathbf{n}\right)$ are evaluated using Roe's averaging, as well as the matrix of left, $\mathbf{T}^{-1}$, and right $\mathbf{T}$, eigenvectors, remembering that $\mathbf{T}$ is the modal matrix that diagonalizes the matrix $\mathbf{A}$. The turbulent speed of sound a_t is computed as $a_t=\sqrt{a^2+\frac{2}{3}\gamma k}=\sqrt{\gamma\frac{p_{eff}}{\rho}}$, using the effective pressure instead of the standard one.

However, the standard Roe's scheme introduces no dissipation for characteristic variables whose eigenvalue is zero, leading to low stability in such cases. To overcome this problem the Harten's entropy-fix, which introduces a lower bound on the eigenvalues, has been introduced in the scheme [104, 105]. In particular, to limit the eigenvalues, we define α_i as follows

$$\alpha_i=\begin{cases}\frac{|\lambda_i|^2+\delta^2}{2\delta} & |\lambda_i|\leq\delta\\ |\lambda_i| & \text{otherwhise}\end{cases}\tag{3.6}$$

with

$$\delta=\delta_{efix}\lambda_{max}.$$

The paramenter $0\leq\delta_{efix}\leq 1$ is called entropy-fix fraction and it is typically set equal to 0.1. For $\delta_{efix}=0$ we have the standard Roe's scheme, for $\delta_{efix}=1$ we obtain a scheme which is close to the Lax-Friedrichs numerical flux. This entropy-fix has the advantage of being differentiable at the point $|\lambda_i|=\delta$.

3.3 Boundary Treatment

3.3.1 Boundary Conditions

Numerical flow simulations take into account only a limited part of the physical domain. Thus, the truncation of the computational domain creates artificial

boundaries, where values of the physical quantities need to be specified. The main problem when imposing such boundary conditions is that the solution on the truncated domain should be as close as possible to a solution which would be obtained for the whole physical domain. Some difficulties can result when the numerical boundary does not coincide with the physical one. The implementation of boundary conditions at artificial boundaries plays an important role for numerical simulation. Most of investigations on implementing boundary conditions have focused on minimizing reflection of spurious waves at the boundaries and applying accurate inflow and outflow conditions to have a robust and efficient solution algorithm. Generally, in standard numerical schemes for simulating compressible flows, the implementation of boundary conditions at artificial boundaries is based on the characteristic variables or the one-dimensional Riemann invariants. This kind of boundary conditions is more appropriate for computing moderate and high Mach number flows, whereas one drawback of this approach is that the Riemann invariants depend on the speed of sound. As a result, at low Mach number flows, the speed of sound approaches infinity and this treatment for boundary conditions may not yield optimum performance [92]. Different kind of boundary conditions, which are capable of absorbing ongoing waves at the artificial boundary, can be found in [47, 50], while non-reflectiong boundary conditions can be found in [51, 63, 94]. We consider the following types of conditions:

Farfield

A simplified set of farfield conditions is used.

- *no vortex correction:* at the subsonic inflow the pressure is taken from the flow field, whereas the other variables are prescribed based on the freestream values

$$\mathbf{u}^b = \left(\rho_\infty, \rho u_{j\infty}, \frac{p\left(\mathbf{u}\right)}{\gamma - 1} + \frac{1}{2}\rho_\infty \mathbf{v}_\infty^2, \rho k_\infty, \rho \omega_\infty \right)^T,$$

with the pressure $p(\mathbf{u})$ computed using the equation of state of a perfect gas. Conversely, at the subsonic outflow the pressure is prescribed based on the freestream value and all the other variables are extrapolated from the flow field

$$\mathbf{u}^b = \left(\rho, \rho u_j, \frac{p_{out}}{\gamma - 1} + \frac{1}{2}\rho \mathbf{v}^2, \rho k, \rho\omega\right)^T.$$

- *vortex correction:* for aerodynamic computations, to minimize issues associated with effect of the farfield boundary and to reduce the distance of the far-field from the wall, which can have influence on the drag and lift computations at high-lift conditions, a vortex correction boundary condition can be used. In this case, density, pressure and velocity are perturbed at the far-field according to [106]. Specifically, at the subsonic inflow the pressure is taken from the flow field and all the other variables are based on perturbed density and velocity

$$\mathbf{u}^b = \left(\rho_\infty, \rho u_{j_\infty}, \frac{p(\mathbf{u})}{\gamma - 1} + \frac{1}{2}(\rho_\infty + \delta\rho)(\mathbf{v}_\infty + \delta\mathbf{v})^2, \rho k_\infty, \rho\omega_\infty\right)^T.$$

Conversely, at the subsonic outflow the pressure is set equal to the perturbed outflow pressure $p_{out} + \delta p$ and all the other variables are taken from the interior

$$\mathbf{u}^b = \left(\rho, \rho u_j, \frac{(p_{out} + \delta p)}{\gamma - 1} + \frac{1}{2}\rho \mathbf{v}^2, \rho k, \rho\omega\right)^T.$$

The values of perturbed density and pressure are given by

$$\delta\rho = -\frac{\rho_\infty}{c_\infty^2}\mathbf{v}_\infty \delta\mathbf{v}, \qquad \delta p = c_\infty^2 \delta p,$$

and the perturbed velocity by

$$\delta\mathbf{v} = \frac{1}{2\pi\rho_\infty|\mathbf{v}_\infty|}\left(\left(L\frac{\beta\eta}{\hat{r}^2} - D\frac{\xi}{\beta\hat{r}^2}\right)\hat{\mathbf{e}}_\xi - \left(L\frac{\beta\xi}{\hat{r}^2} + D\frac{\beta\eta}{\hat{r}^2}\right)\hat{\mathbf{e}}_\eta\right),$$

where D and L are the drag and lift forces, $\hat{\mathbf{e}}_\xi$ and $\hat{\mathbf{e}}_\eta$ are the freestream aligned and normal unit vectors respectively, $\xi = (\mathbf{x} - \mathbf{x}_0)\hat{\mathbf{e}}_\xi$ and $\eta = (\mathbf{x} - \mathbf{x}_0)\hat{\mathbf{e}}_\eta$ with $\mathbf{x}_0$ origin of the vortex, $\beta = \sqrt{1 - M_\infty}$ and $\hat{r} = \sqrt{\xi^2 + \beta^2\eta^2}$.

Wall

- **Adiabatic wall**

  At the viscous wall a set of adiabatic boundary conditions is employed

  $$\mathbf{u}^b = (\rho, 0, 0, \rho E, 0, \rho\omega_w)\,,$$

  where ω_w is computed following a modified version of the Menter's approach. In detail, the standard Menter's boundary condition [80]

  $$\omega_w = \frac{6\nu}{\beta(\alpha_p y_1)^2},$$

  with $\nu = \frac{\mu}{\rho}$ kinematic viscosity, y_1 first cell-centroid height and $\alpha_p = \frac{1}{\sqrt{10}}$ is replaced by a modified version proposed by Schoenawa and Hartmann [82] which takes into account of the degree of the polynomial approximation of the numerical solution. For this aim, α_p is calculated by projecting the near wall solution of $\omega\,(y) = \frac{6\nu}{\beta y^2}$ to the polynomial space of order p employed.

Symmetry

- Symmetry boundary conditions should be defined such that the discretization on the computational domain, called half domain, resembles the discretization on the full domain, defined as the half domain plus its mirror with respect to the symmetry boundary. The state $\mathbf{u}^b$ has the opposite normal velocity component $(\mathbf{v}\cdot\mathbf{n}) = (\mathbf{v}\cdot\mathbf{n})^+$ whereas the other variables are prescribed based on the interior. In this way the non-permeability condition is satisfied and the mass flux computed by the Riemann solver is null. Also the gradient $\nabla\mathbf{u}_h{}^b$ is modified. For a more detailed description refer to [83].

3.3.2 Boundary representation

In many applications the domain Ω is not a polygonal domain but it includes curved boundaries, therefore the boundary cannot be accurately represented by the triangulation τ_h. It has been shown by Bassi and Rebay in [64] that the DG method is highly sensitive to the accuracy of the boundary representation and errors due to geometrical approximation may dominate the discretization error, rendering the use of a high-order schemes useless. Therefore, a higher order approximation of the domain boundary, achieved by using mesh elements with one or more curved sides usually described by polynomials, is mandatory to compute accurate solutions. In this work, we consider polynomial mapping functions, expressed in terms of Lagrangian basis function and Lagrangian support points (nodes). Thus, the mapping between the canonical triangle or square and the element in physical space is given by

$$\mathbf{x} = \sum_j \mathbf{x}^{(j)} \phi_j \left(\xi \right), \tag{3.7}$$

where ϕ_j is the jth basis function, ξ is the location in the reference space, and $\mathbf{x}^{(j)}$ is the location of the jth node in physical space. In general, due to the higher order boundary approximation, the Jacobian of the mapping is not constant. This means that triangles and quadrangles with curved edges are allowed. Finally, by placing the boundary nodes on the real domain boundary, a higher order geometry representation is achieved.

3.4 Time integration

The DG space discretization, Eq. (3.2), results in the following system of ordinary differential equations in time

$$\mathbf{M}_\Gamma \frac{d\mathbf{U}}{dt} + \mathbf{R} \left(\mathbf{U} \right) = 0, \tag{3.8}$$

where $\mathbf{M}_\Gamma$ denotes the global system matrix, $\mathbf{U}$ and $\mathbf{R}$ are the global vectors of degrees of freedom (dofs) and of residuals, respectively. In this work, the

semidiscrete system of Eq. (3.8) is advanced in time to the steady state using the implicit backward Euler scheme:

$$\left[\frac{\mathbf{M}_\Gamma}{\Delta t} + \frac{\partial \mathbf{R}^{\mathrm{n}}}{\partial \mathbf{U}}\right] \Delta \mathbf{U}^{\mathrm{n}} = -\mathbf{R}^{\mathrm{n}}, \tag{3.9}$$

where $\Delta \mathbf{U}^{\mathrm{n}} = \mathbf{U}^{\mathrm{n}+1} - \mathbf{U}^{\mathrm{n}}$, $\frac{\partial \mathbf{R}^{\mathrm{n}}}{\partial \mathbf{U}}$ is the Jacobian matrix of the DG space discretization and $\left[\frac{\mathbf{M}_\Gamma}{\Delta t} + \frac{\partial \mathbf{R}^{\mathrm{n}}}{\partial \mathbf{U}}\right]$ denotes the global system matrix. The matrix $\left[\frac{\mathbf{M}_\Gamma}{\Delta t} + \frac{\partial \mathbf{R}^{\mathrm{n}}}{\partial \mathbf{U}}\right]$ can be regarded as an $N_K \times N_K$ block sparse matrix where N_K is the number of elements in τ_h and the rank of each block is $m \times N_{dof}^K$, where N_{dof}^K is the number of dofs for each of the m conservative variables in the generic element K. The Jacobian matrix of the DG discretization has been computed analytically without any approximation, except of the dissipative part of the numerical flux for which the Jacobian is computed numerically. Using very large time steps, the method can therefore achieve quadratic convergence in the computation of steady state solutions. For the backward Euler scheme and in the limit $\Delta t \to \infty$ Eq. (3.9) is identical to one iteration of the Newton method applied to the steady discrete problem. Finally, we remark that we have used the restarted GMRES algorithm with ILU(0) available in the PETSc [100] library to solve Eq. (3.9).

Chapter 4

Low Mach number preconditioning

In this chapter we present the preconditioning of the fully coupled RANS $k-\omega$ equations in order to improve the accuracy and efficiency of the solution at low Mach numbers using an implicit time integration method. Unlike the flux preconditioning approach, that modifying the dissipative terms of the numerical flux function could also be used for time accurate computations [103], the here employed full preconditioning approach, that alters both the time-derivative terms of the governing equations and the numerical flux function, can be applied to steady-state simulations only.

4.1 The preconditioning approaches with implicit schemes

The time dependent system of the RANS $k-\omega$ equations becomes very stiff at low Mach number. Indeed the condition number, defined as the ratio between the largest and smallest wave speeds, $N_{cond} = \frac{|\lambda_{max}|}{|\lambda_{min}|} = \frac{|\mathbf{v}\cdot\mathbf{n}+a_t|}{|\mathbf{v}\cdot\mathbf{n}|}$, increases without bound since the smallest wave speed approaches zero.

It's well-known that explicit schemes are typically subject to restrictive limitations on the CFL number, and they are particularly slow to converge at low Mach numbers. Conversely, implicit schemes as compared to explicit ones allow to use significantly larger time steps without hampering the stability of the time integration process. Moreover, they have superior robustness and convergence speed in case of stiff equation systems, which are often encountered in real gas simulations, turbulent computations and highly stretched grids. Otherwise, the computational effort per time step or iteration and the required memory of implicit schemes are significantly higher than those required by explicit ones. Furthermore implicit schemes are also more difficult to be implemented.

Despite implicit schemes may be more suitable for low-speed flows [108], they are still adversely affected by stiffness. As a result, low Mach number preconditioning is recommended to guarantee the efficiency as well as the accuracy of numerical solution in case of upwind schemes [30]. For steady-state computations the most efficient approach is represented by the full preconditioning, which is based on both the alteration of the time derivative terms of the governing equations and of the dissipative terms of the numerical flux function. The premultiplication of the unsteady terms of the RANS system by the low Mach number preconditioning matrix equalizes the speeds of convective and acustic waves, improving the convergence without altering the steady state solution, whereas the preconditioning of numerical fluxes operates so that the dissipation terms scale properly with the convective ones,

improving the accuracy of the solution at low Mach number [30]. Conversely, to compute efficiently and accurately unsteady low Mach number flows, the flux preconditioning approach, which modifies only the dissipative terms of the numerical flux, can be used. This formulation is consistent in time since the time-derivative terms of the governing equations are not changed, it is quite simple to implement in existing implicit DG codes and it ensures both accuracy and efficiency improvements since it affects both numerical fluxes and Jacobian of the residuals [52, 103]. Otherwise, time-accurate preconditioned governing equations can be solved by means of a dual time-stepping approach, where a modified steady problem is solved at each physical time step by advancing in pseudo time [48].

4.1.1 Low Mach number preconditioned RANS system

The preconditioned RANS and $k-\omega$ equations in conservative form are obtained from Eq. (2.1) by replacing $\mathbf{\Gamma}$ with the preconditioning matrix, $\mathbf{P}_u$, expressed in conservative variables:

$$\mathbf{P}_u \frac{\partial \mathbf{u}}{\partial t} + \boldsymbol{\nabla} \cdot \mathbf{F}_c(\mathbf{u}) - \boldsymbol{\nabla} \cdot \mathbf{F}_v(\mathbf{u}, \boldsymbol{\nabla}\mathbf{u}) - \mathbf{S}(\mathbf{u}, \boldsymbol{\nabla}\mathbf{u}) = \mathbf{0}. \tag{4.1}$$

The preconditioning matrix used in this work is based on a modified version of the preconditioner proposed by Turkel in [57]. Given $\mathbf{P}_q$, the preconditioning matrix in primitive variables

$$\mathbf{P}_q = \begin{pmatrix} \frac{1}{\beta^2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ -\frac{\left(\beta^2-1\right)\left(\frac{2}{3}\rho_p k+1\right)}{\left(\rho C_p-\frac{2}{3}\rho_T k\right)\beta^2} & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{4.2}$$

with

$$\rho_p = \left.\frac{\partial \rho}{\partial p}\right|_{T=const.} = \frac{1}{T}, \quad \rho_T = \left.\frac{\partial \rho}{\partial T}\right|_{p=const.} = -\frac{p}{T^2},$$

$\mathbf{P}_u$ is computed by means of the Jacobians of variables transformation [102] as follows,

$$\mathbf{P}_u = \frac{\partial \mathbf{u}}{\partial \mathbf{q}} \mathbf{P}_q \frac{\partial \mathbf{q}}{\partial \mathbf{u}}.$$

With respect to the standard Turkel's preconditioning matrix, the entry $\mathbf{P}_q(4,1)$ has been changed to take into account the proposed formulation of the governing equations [45]. In particular, the preconditioner has been modified without altering the eigenvalues' multiplicity and structure with respect to the non-preconditioned case. Note that, if the preconditioning is applied to the non-conservative formulation of the governing equations the preconditioner, $\mathbf{P}_q$, only modifies the continuity and energy equations. Instead when the conservative form is chosen, continuity and energy equations are coupled with

momentum and turbulence model equations through the more complex preconditioner, $\mathbf{P}_u$, obtained by multiplying $\mathbf{P}_q$ by the transformation matrices $\frac{\partial \mathbf{u}}{\partial \mathbf{q}}$ and $\frac{\partial \mathbf{q}}{\partial \mathbf{u}}$.

The low Mach number preconditioning is devoted to eliminate the ill-conditioning of the system of equations as the Mach number tends to zero. For this purpose, the preconditioning parameter β, which controls the wave speeds of the system, should be chosen as $O(M)$, enabling to achieve a bounded condition number when the local Mach number M of the flow tends to zero. However, near stagnation points, M, and hence β, become very small, making the preconditioning matrix close to be singular. As suggested by Turkel [57], to prevent singularities at stagnation points we introduce the following cut-off

$$\beta^2 = min\left(max\left(M^2, \varepsilon {M_\infty}^2\right), 1\right). \tag{4.3}$$

Here ε is a problem dependent parameter whose choice affects both the convergence rate and the accuracy of the converged solution through its impact on the dissipation matrix [109]. Moreover, it is a critical factor influencing the robustness of the scheme. Indeed, as shown by Colin *et al.* [90] and Choi and Merkle [49], the major shortcoming of the low Mach number preconditioning is a reduced robustness, which is due, for Euler computations, to stagnation point regions that result in high local pressure disturbances, whereas, for turbulent problems, to low Reynolds number regions that may lead to instabilities. Specifically, for viscosity dominated flows, the boundary layer region is dominated by diffusion processes. Numerical studies pointed out the need to decrease the preconditioning effect in such critical regions in order to preserve the robustness of the numerical scheme [49]. This can be achieved by setting properly values of ε. Generally, the criterion to set ε is that it should be chosen as small as possible to preserve the accuracy of the numerical solution, but large enough for not hampering the efficiency. Typically ε is set equal to 1 for viscous and easy turbulent flows, whereas it can be increased, for turbulent flows and complex geometries [57, 109]. It's worth to note that choosing a

large ε implies that β is quite constant through the domain. Several additional limitations have been proposed for viscous computation in order to limit the reference velocity such that it does not become smaller than the local diffusion velocity [48, 90]. Nevertheless, our numerical experiments on external flows showed that the above limitation of Eq. (4.3) allowed to obtain very accurate DG solutions with a good level of robustness without having to resort to additional viscous limitations in agreement with results presented by Unrau and Zingg [109] for the FV discretization.

Finally, in order to preserve the accuracy and performance of the method for solving high-speed flows the preconditioner smoothly reduces to the identity matrix, recovering the non-preconditioned system of equations and the standard Roe's scheme. Specifically, in Eq. (4.3), $\beta \to 1$ as $M \to 1$ and $\beta = 1$ for $M \geq 1$.

The resultant eigenvalues of the preconditioned system Eq. (4.1) are given by

$$\hat{\mathbf{\Lambda}} = \mathrm{diag}\left\{\mathbf{v}\cdot\mathbf{n}, \mathbf{v}\cdot\mathbf{n}, \mathbf{v}\cdot\mathbf{n}, \mathbf{v}\cdot\mathbf{n}, v' + a'_t, v' - a'_t\right\},$$

where

$$v' = \frac{1}{2}\left(\beta^2 + 1\right)(\mathbf{v}\cdot\mathbf{n}),$$
$$a'_t = \frac{1}{2}\sqrt{(\mathbf{v}\cdot\mathbf{n})^2(\beta^2-1)^2 + 4\beta^2 {a_t}^2}.$$

and a_t is the turbulent sound speed. Note that the choice a_t=a yields the preconditioned eigenvalues found by Turkel [57]. We remark that only the acustic waves are changed by low Mach number preconditioning. At low speed, $\beta \to 0$ as $M \to 0$ and the condition number of the preconditioned system, $N_{cond} = \frac{|v'+a'_t|}{|\mathbf{v}\cdot\mathbf{n}|}$, tends to 1, demonstrating the well-conditioning of the proposed preconditioned system in the incompressible limit.

4.1.2 Preconditioned numerical flux function

To extend the Roe's numerical flux function to the incompressible limit we replace the dissipation matrix in Eq. (3.5) with a preconditioned version as

follows,

$$\mathbf{H}\left(\mathbf{u}^{+},\mathbf{u}^{-},\mathbf{n}\right)=\frac{1}{2}\left(\mathbf{F}_{c}(\mathbf{u}^{+})\cdot\mathbf{n}+\mathbf{F}_{c}(\mathbf{u}^{-})\cdot\mathbf{n}-\hat{\mathbf{A}}\left(\mathbf{u}^{+},\mathbf{u}^{-},\mathbf{n}\right)\Delta\mathbf{u}\right).$$

The preconditioned dissipation term is given by

$$\hat{\mathbf{A}}\left(\mathbf{u}^{+},\mathbf{u}^{-},\mathbf{n}\right)\Delta\mathbf{u}=\mathbf{P}_{u}\left|{\mathbf{P}_{u}}^{-1}\left(\frac{\partial\mathbf{F}_{c}}{\partial\mathbf{u}}\cdot\mathbf{n}\right)\right|\Delta\mathbf{u}=$$

$$=\mathbf{P}_{u}\Big(\ \hat{\mathbf{T}}|\hat{\mathbf{\Lambda}}|\hat{\mathbf{T}}^{-1}\Big)\Delta\mathbf{u},$$

where $\hat{\mathbf{\Lambda}}=\mathrm{diag}\left\{\mathbf{v}\cdot\mathbf{n},\mathbf{v}\cdot\mathbf{n},\mathbf{v}\cdot\mathbf{n},\mathbf{v}\cdot\mathbf{n},v'+a'_{t},v'-a'_{t}\right\}$ and $\hat{\mathbf{T}}$ diagonalizes the preconditioned Jacobian ${\mathbf{P}_{u}}^{-1}\left(\frac{\partial\mathbf{F}_{c}}{\partial\mathbf{u}}\cdot\mathbf{n}\right)$. In order to simplify the eigenvector decomposition it is easier to operate using the set of primitive variables and to convert the dissipative terms to conservative form by means of the transformation matrix $\frac{\partial\mathbf{u}}{\partial\mathbf{q}}$ [52]. Hence, the dissipation term can be computed as

$$\hat{\mathbf{A}}\left(\mathbf{u}^{+},\mathbf{u}^{-},\mathbf{n}\right)\Delta\mathbf{u}=\frac{\partial\mathbf{u}}{\partial\mathbf{q}}\mathbf{P}_{q}\left|{\mathbf{P}_{q}}^{-1}\frac{\partial\mathbf{q}}{\partial\mathbf{u}}\left(\frac{\partial\mathbf{F}_{c}}{\partial\mathbf{q}}\cdot\mathbf{n}\right)\right|\Delta\mathbf{q}=\frac{\partial\mathbf{u}}{\partial\mathbf{q}}\mathbf{P}_{q}\Big(\ \hat{\mathbf{T}}_{q}|\hat{\mathbf{\Lambda}}|\hat{\mathbf{T}}_{q}^{-1}\Big)\Delta\mathbf{q}.$$

Here, $\hat{\mathbf{T}}_{q}$ is the left eigenvector matrix which diagonalizes the preconditioned Jacobian in primitive variables ${\mathbf{P}_{q}}^{-1}\frac{\partial\mathbf{q}}{\partial\mathbf{u}}\left(\frac{\partial\mathbf{F}_{c}}{\partial\mathbf{q}}\cdot\mathbf{n}\right)$. The subscript $\mathbf{q}$ denotes that the modal matrix $\hat{\mathbf{T}}_{q}$ refers to the Jacobian in primitive variables. Note that, the eigenvectors may become singular for some values of the numerical parameters used for their definition, i.e. $\mathbf{n}=(n_{1},n_{2})$. Nevertheless, since there is a certain flexibility in the choice of the set of eigenvectors, these have been determined in such a way as to be always non-vanishing independent, see Appendix B.

Finally, also in the case of the preconditioned numerical flux function the entropy fix can be applied to the Roe's scheme, limiting the preconditioned eigenvalues such as

$$\hat{\mathbf{\Lambda}}=\mathrm{diag}\left\{\hat{\alpha}_{i}\right\},$$

with $\hat{\alpha}_{i}$ computed according to the formula given in Eq. (3.6).

4.1.3 Boundary conditions

In general the boundary conditions at the far-field are based on characteristic variables, even for viscous flows. Since the time-dependent terms of the governing equations are changed by preconditioning, also the characteristics of the resulting system are modified, although the signs of the eigenvalues remain unchanged. Therefore, the boundary conditions at far-field need to be modified accordingly [94]. Nevertheless, it was shown in [57, 92, 93] that a simplified set of boundary conditions based on the specification of density and velocity at the inflow and pressure at the outflow is generally non-reflective or weakly reflective in combination with low Mach number preconditioning.

It's worth to note that if the flux preconditioning approach is applied, the characteristics of the system remain unchanged and the non-preconditioned boundary conditions can be set at the far-field.

At last, we remark that the preconditioning has no effect at the wall, where zero heat flux no-slip boundary conditions are imposed. Thus, the same wall boundary conditions can be employed for both the preconditioned and non-preconditioned schemes.

4.2 Numerical Results

In this section we present a series of results obtained for the following steady test cases: laminar flow around a NACA0012 airfoil, turbulent flow over a flat plate and three-element L1T2 high-lift configuration. The computations have been performed at low Mach number, with and without preconditioning, using various degrees of polynomial approximations. The first test case shows the performance of the proposed DG method in solving laminar flow at different low Mach numbers. A comparison between flux and full preconditioning approach is also presented. The last two test cases will demonstrate the capabilities of the full preconditioning technique in conjunction with an implicit scheme in solving turbulent flows.

4.2.1 Laminar flow around a NACA0012 Airfoil

In this first numerical experiment we consider a low Mach number laminar flow around a NACA0012 airfoil. For this test case the simulations have been carried out using the `MIGALE` code, which uses primitive variables for the computations. Two grid topologies (quadrangular and triangular) have been used in order to investigate the behaviour of both the standard and the preconditioned DG scheme for different spatial discretizations, see Figure 4.1. The quadrangular grid is a C-type grid with 1792 elements, the triangular grid with 3584 elements results from the triangulation of the quadrangular one. Curved boundaries have been approximated by piecewise cubic elements according to Bassi and Rebay [65]. The distance of the far-field boundary from the profile of unit (chord) length is about 55 chords. At far-field we specify the Mach number, M_∞, at a zero angle of attack, with Reynolds number $Re_\infty = 500$. On the wall boundary of the airfoil we impose a zero heat flux (adiabatic) no slip boundary condition. The simulations have been performed at $M_\infty = 10^{-1}$, $M_\infty = 10^{-2}$ and $M_\infty = 10^{-3}$, using linear (P_1), quadratic (P_2) and cubic elements (P_3). The convergence of the solution process is presented in terms of

the normalized L_2-norm of the residuals versus the number of iterations and versus the CPU time, whereas the accuracy of the numerical solution is analyzed, from a qualitative point of view, by the contour plots of the normalized pressure (defined as $p_{norm} = (p - p_{min})/(p_{max} - p_{min})$).

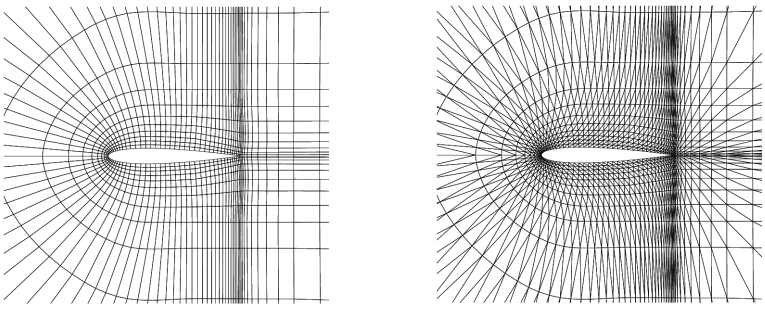


Figure 4.1: Computational Grids for NACA0012 test case: quadrangular (left) and triangular (right).

Effects of flux preconditioning approach on convergence speed

The convergence histories are shown only for the quadrangular grid as similar histories are obtained on the triangular one. The computations are performed for a number of Krylov subspace vectors = 120, number of restarts = 1 and relative tolerance to stop iterative solution = 10^{-6}. Figure 4.2 compares the history of residuals versus the number of implicit iteration steps of the backward Euler scheme with and without flux preconditioning. The plots show a deterioration in the convergence rate without preconditioning as the Mach number gets smaller whilst the preconditioned scheme always produces quadratic convergence. The effect is appreciable at $M_\infty = 10^{-2}$ and more evident at $M_\infty = 10^{-3}$. At $M_\infty = 10^{-1}$ both the non-preconditioned and preconditioned DG schemes converge at about the same convergence rate independently of the polynomial degree. Furthermore the corresponding non-preconditioned and preconditioned residual histories decrease about the same

order of magnitude. In particular, the residuals of velocity components are indistinguishable, whereas the non-preconditioned residuals of pressure and temperature present a slightly larger decrease than the corresponding preconditioned one. This behaviour becomes more evident at $M_\infty = 10^{-2}$, and at $M_\infty = 10^{-3}$ because of round-off errors. The major benefits of preconditioning technique are shown at $M_\infty = 10^{-3}$, for quadratic and cubic elements, where the convergence of the numerical solution is not reached without preconditioning. This is due to the effect of low Mach number preconditioning on the linear system matrix, through the Jacobian of the residuals. In particular, with preconditioning the full convergence of the residuals is reached quadratically in about 10 iterations, independently of both Mach number and polynomial degree, like for the inviscid case [66]. Figure 4.3 compares the history of residuals versus CPU time (seconds), computed on the quadrangular grid with and without flux preconditioning. The plots show that, using the non-preconditioned Roe's flux, the overhead in terms of CPU time increases as the Mach number gets smaller and the polynomial degree increases, whilst it is almost independent of the Mach number with flux preconditioning.

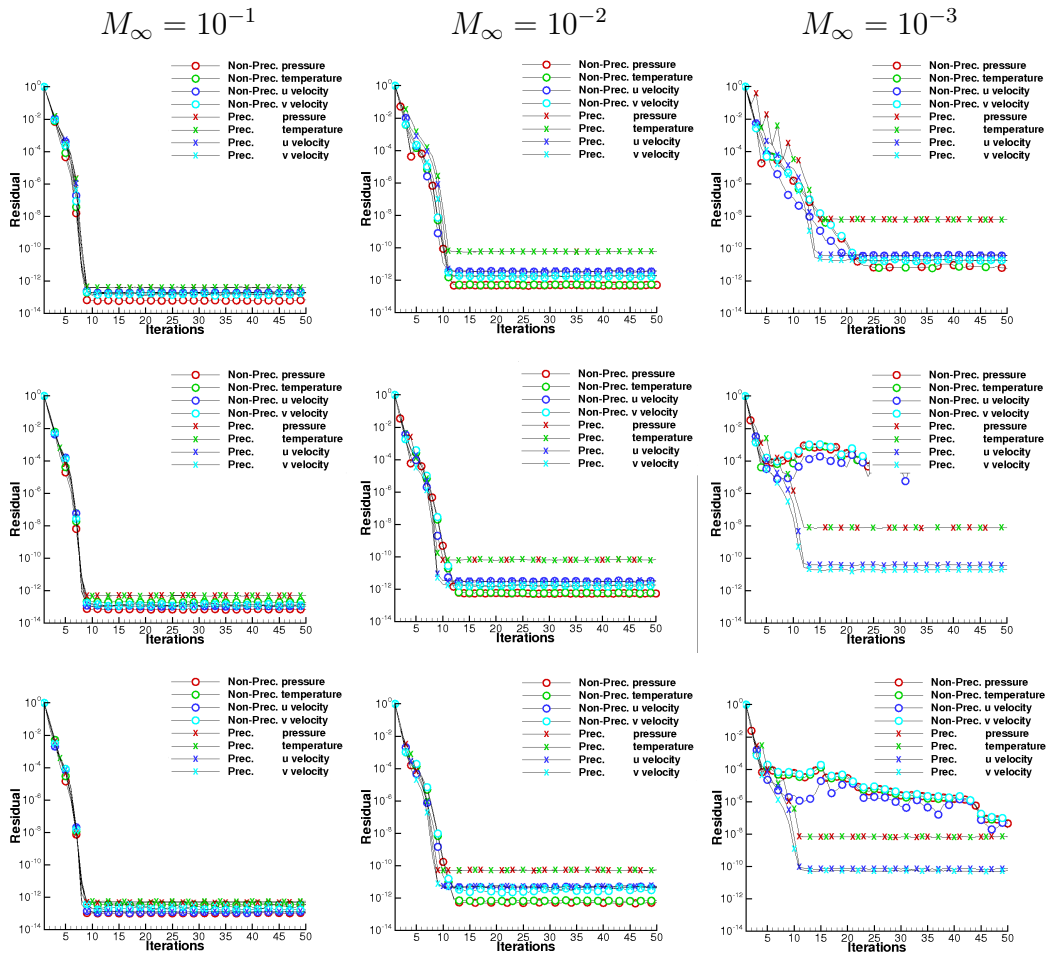


Figure 4.2: History of L_2-norm of residuals versus number of iterations for the quadrangular grid. $M_\infty = 10^{-1}$(left column), $M_\infty = 10^{-2}$ (middle column) and $M_\infty = 10^{-3}$ (right column). Linear, P_1 (top row), quadratic, P_2 (middle row) and cubic, P_3 (bottom row) elements.

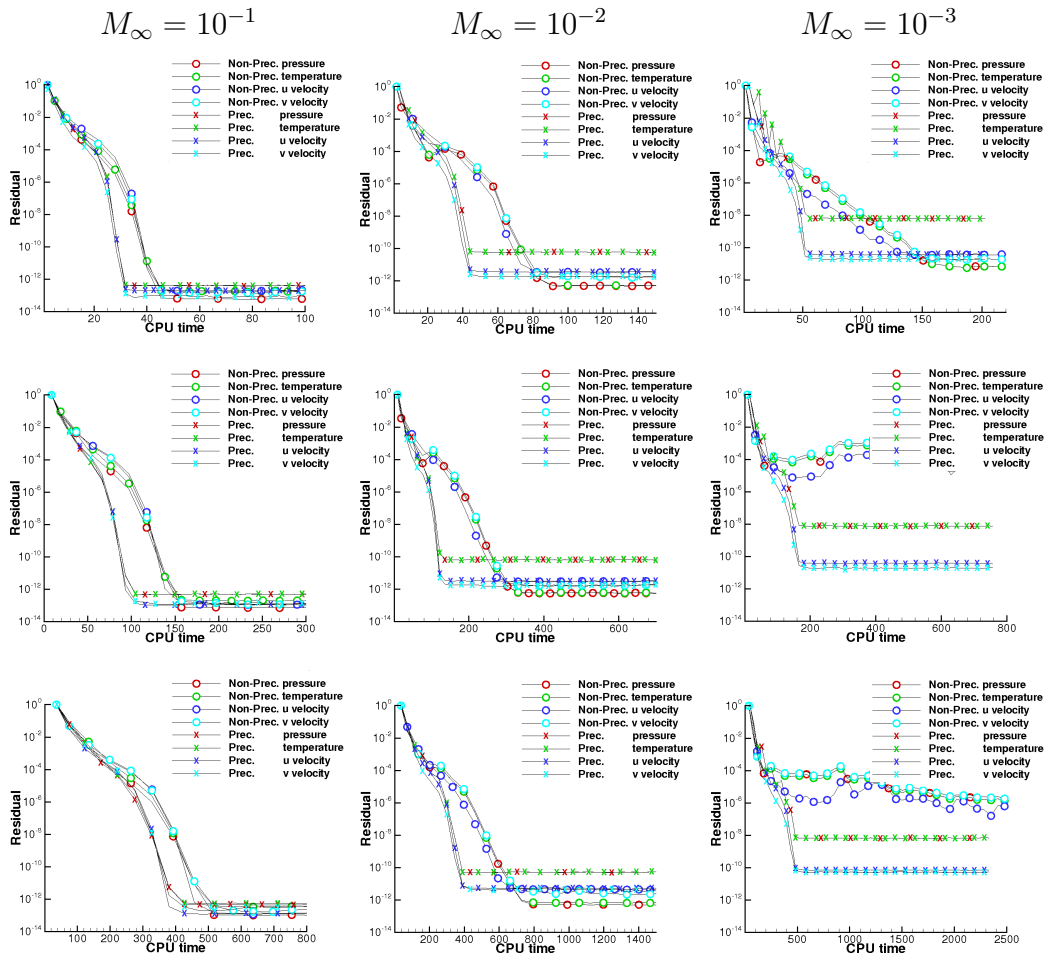


Figure 4.3: History of L_2-norm of residuals versus CPU time for the quadrangular grid. $M_\infty = 10^{-1}$(left column), $M_\infty = 10^{-2}$ (middle column) and $M_\infty = 10^{-3}$ (right column). Linear, P_1 (top row), quadratic, P_2 (middle row) and cubic, P_3 (bottom row) elements.

The figure 4.4 illustrates the performance of the GMRES solver with (right column) and without (left column) low Mach number preconditioning. The results refer to linear, quadratic and cubic elements at $M_\infty = 10^{-2}$. Similar results have been observed for $M_\infty = 10^{-1}$ and $M_\infty = 10^{-3}$. The plots on the top row show the number of GMRES iterations (open symbols) and the logarithm of CFL number (solid symbols), while those on the bottom row show the ratio between the L_2-norms of the last and the first residual of the

GMRES iterative solution. The quantity along the x-axis is the number of non-linear iterations. We can observe that increasing the CFL number the computations performed without low Mach number preconditioning rapidly use up the maximum number of GMRES iterations (240) without satisfying the required six-order drop of residuals. Instead the low Mach number preconditioning improves the efficiency of GMRES solver so that the preconditioned solutions require somewhat less than 120 GMRES iterations to solve the linear system within each time step, even for the highest CFL numbers. We notice that in comparison to the inviscid case [66] a higher number of Krylov subspace vectors was used (120 instead of 60), with the same number of restarts (1) and relative tolerance (10^{-6}). Finally we mention that, at $M_\infty = 10^{-2}$, the cost to compute the analytical Jacobian relative to the computational cost of a full time step using 240 GMRES iterations is around 31%, 41% and 60% for the P_1, P_2 and P_3 solutions, respectively.

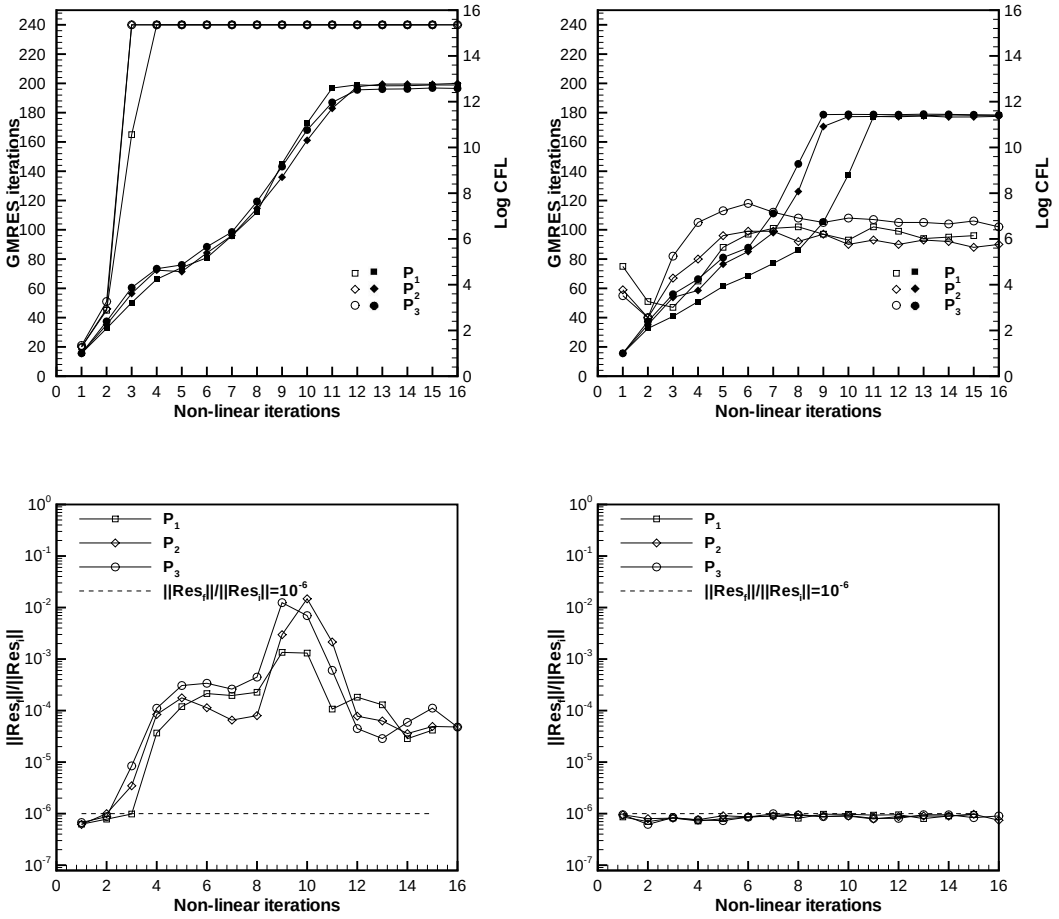


Figure 4.4: Behaviour of the GMRES solver with (right column) and without (left column) low Mach number preconditioning for $M_\infty = 10^{-2}$.

Effects of full preconditioning on convergence speed

In general, preconditioning modifies the unsteady terms of the governing equations by a preconditioning matrix. This results in an improved residual convergence speed of steady-state solutions. Figure 4.5 compares the history of residuals versus the number of iterations (top row) and versus CPU time (bottom row) with and without the time derivative preconditioning for the quadrangular grid. The plots show computations performed at $M_\infty = 10^{-2}$ for linear, quadratic and cubic elements. Similar results have been obtained

at $M_\infty = 10^{-1}$ and $M_\infty = 10^{-3}$. At far-field a set of non-reflecting boundary conditions are employed which specify entropy, stagnation enthalpy, and the tangential velocity at the inlet and pressure at the outlet boundary [92]. The graphs show that the corresponding residual histories obtained with the flux-preconditioning and full-preconditioning approach decrease about the same order of magnitude. Nevertheless the preconditioning of the time dependent terms produces an improvement in the convergence rate and CPU time. For all polynomial approximations, the full convergence was reached in a lower number of non-linear iterations. Furthermore, the overhead, in terms of CPU time, reduces as the polynomial degree increases. The gain of efficiency is due to the improvement of the condition number of the linear system matrix produced by the preconditioning of the time derivative terms. This is evident in Figure 4.6 where we compared the performance of the GMRES solver with flux (open symbols) and full (solid symbols) low Mach number preconditioning. The graphs show the results for P_1, P_2 and P_3 solutions at $M_\infty = 10^{-2}$. The plot on the left column shows the logarithm of CFL number while that on the right column shows the number of GMRES iterations. The quantity on the x-axis is the number of non-linear iterations. We remark that the plots of Figure 4.6 have been obtained for fixed GMRES parameters (number of Krylov-subspace vectors = 120 with one restarts performed at and relative tolerance to stop iterative solution = 10^{-6}). The result of the full preconditioning approach is that it allows higher CFL numbers with a reduced number of linear iterations needed to reach the full convergence of each variable as compared to the flux preconditioning approach.

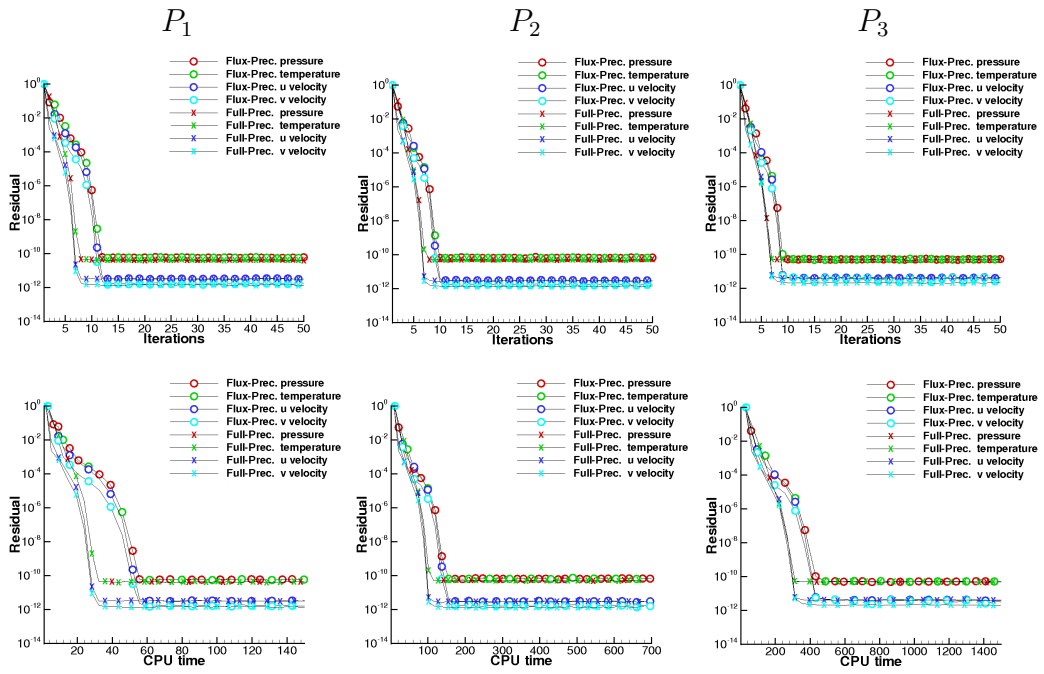


Figure 4.5: History of L_2-norm of residuals versus number of iterations (top row) and CPU time (bottom row) for $M_\infty = 10^{-2}$ (quadrangular grid). Linear, P_1 (top row), quadratic, P_2 (middle row) and cubic, P_3 (bottom row) elements.

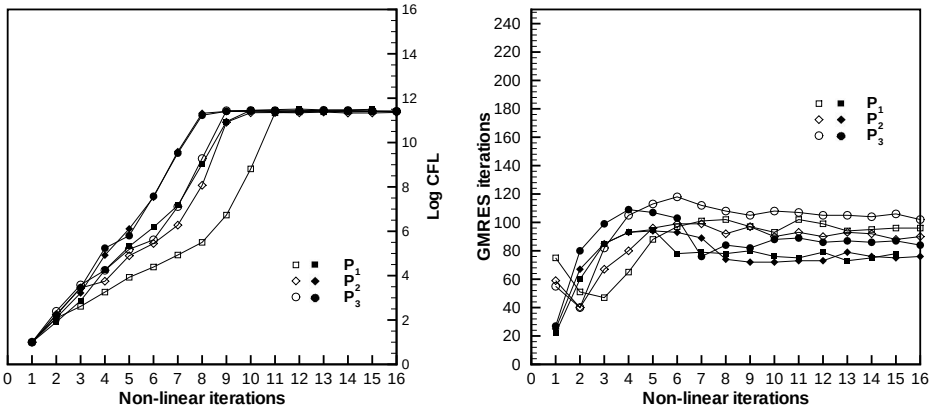


Figure 4.6: Behaviour of the GMRES solver with flux (open symbols) and full (solid symbols) preconditioning: Logarithm of the CFL numbers vs non-linear iterations (left column) and GMRES iterations vs non-linear iterations (right column) at $M_\infty = 10^{-2}$.

Effects of preconditioning on the solution accuracy

The isolines of normalized pressure computed on the quadrangular grid at $M_\infty = 10^{-1}$, $M_\infty = 10^{-2}$, $M_\infty = 10^{-3}$ using linear, quadratic and cubic elements without and with preconditioning are displayed in Figures 4.7 and 4.8, respectively. As expected, the preconditioned solutions are more accurate than the corresponding non-preconditioned ones by virtue of the preconditioning of the dissipative term of the Roe's Riemann solver. We note that for a given Mach number, the loss of accuracy of the non-preconditioned solutions reduces, due to the increased polynomial degree when passing from linear to cubic elements. Conversely, for a given polynomial degree, the non-preconditioned solution becomes worse as the Mach number reduces. A distinguishing feature of this test case is the presence of a stagnation point at the leading edge. In Figure 4.9 we compare the contours of normalized pressure without (top row) and with (middle row) preconditioning, near the leading edge, computed on a quadrangular grid, at $M_\infty = 10^{-3}$, for linear P_1 (left column), quadratic P_2 (central

column), and cubic P_3 (right column) elements. The non-preconditioned contours at $M_\infty = 0.4$ (bottom row) are also shown as reference. It is evident that approaching the stagnation point the solution degrades and that this effect reduces by increasing the degree of polynomial approximation. Figure 4.10 shows the same local analysis of Figure 4.9, for the triangular grid. Overall, it is worth noting that also in the case of the Navier-Stokes equations, the DG discretization on triangular grid yields remarkably accurate solutions at low Mach number even without preconditioning, like found for the inviscid case [66]. In particular, the preconditioned and the non-preconditioned contours of normalized pressure are almost indistinguishable using P_3 elements, whereas some small differences can be seen in the P_1 and P_2 solutions. The marked influence of the geometrical shape of the elements on the accuracy of the Roe's flux in the low Mach number limit could be explained by the asymptotic analysis recently performed by Rieper and Bader [95, 96] and by Guillard [97], for the first-order Roe scheme. Our current work seems to indicate that low order DG schemes face the same problems as the standard finite volume upwind schemes: at low Mach number they only work on triangular elements.

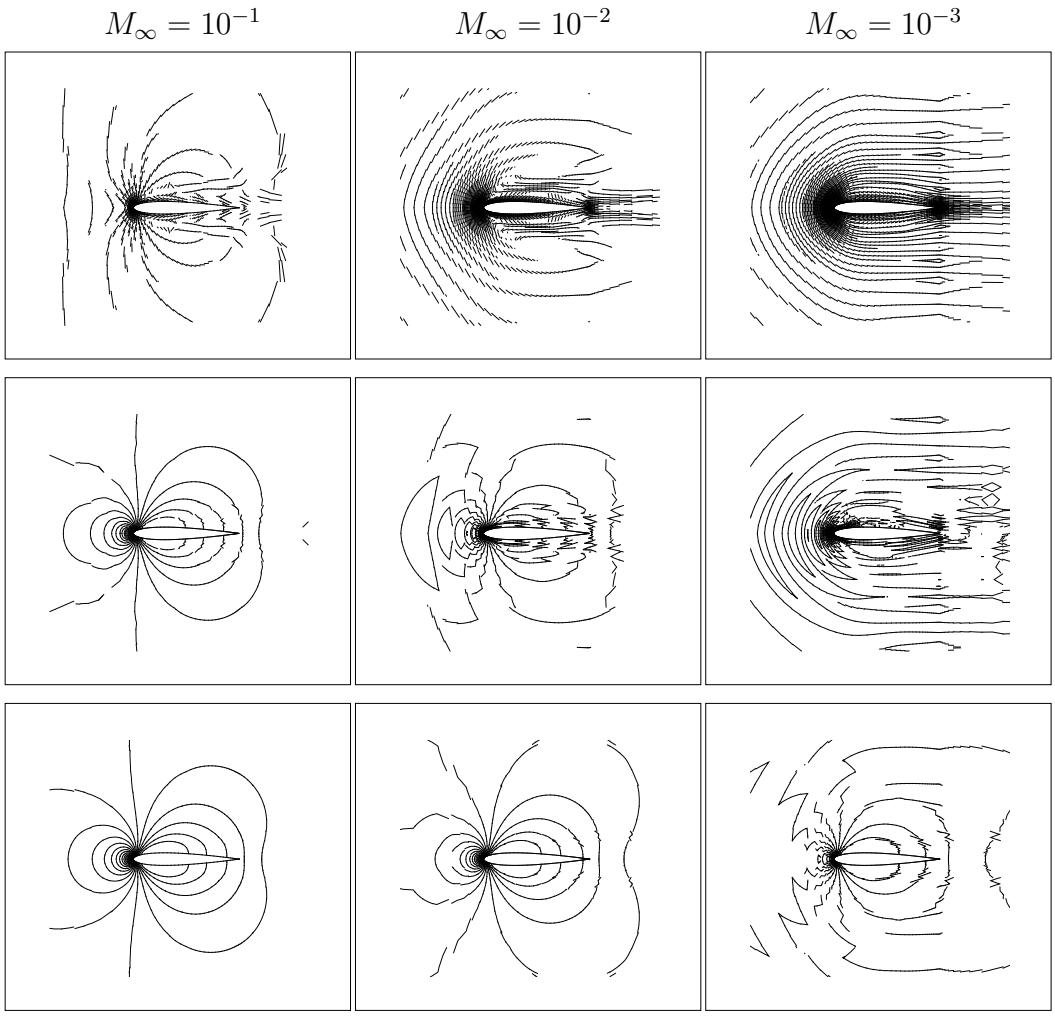


Figure 4.7: NACA0012 test case: contours of normalized pressure without preconditioning for quadrangular grid. $M_\infty = 10^{-1}$ (left column), $M_\infty = 10^{-2}$ (middle column) and $M_\infty = 10^{-3}$ (right column). P_1 (top row), P_2 (middle row) and P_3 (bottom rows) elements.

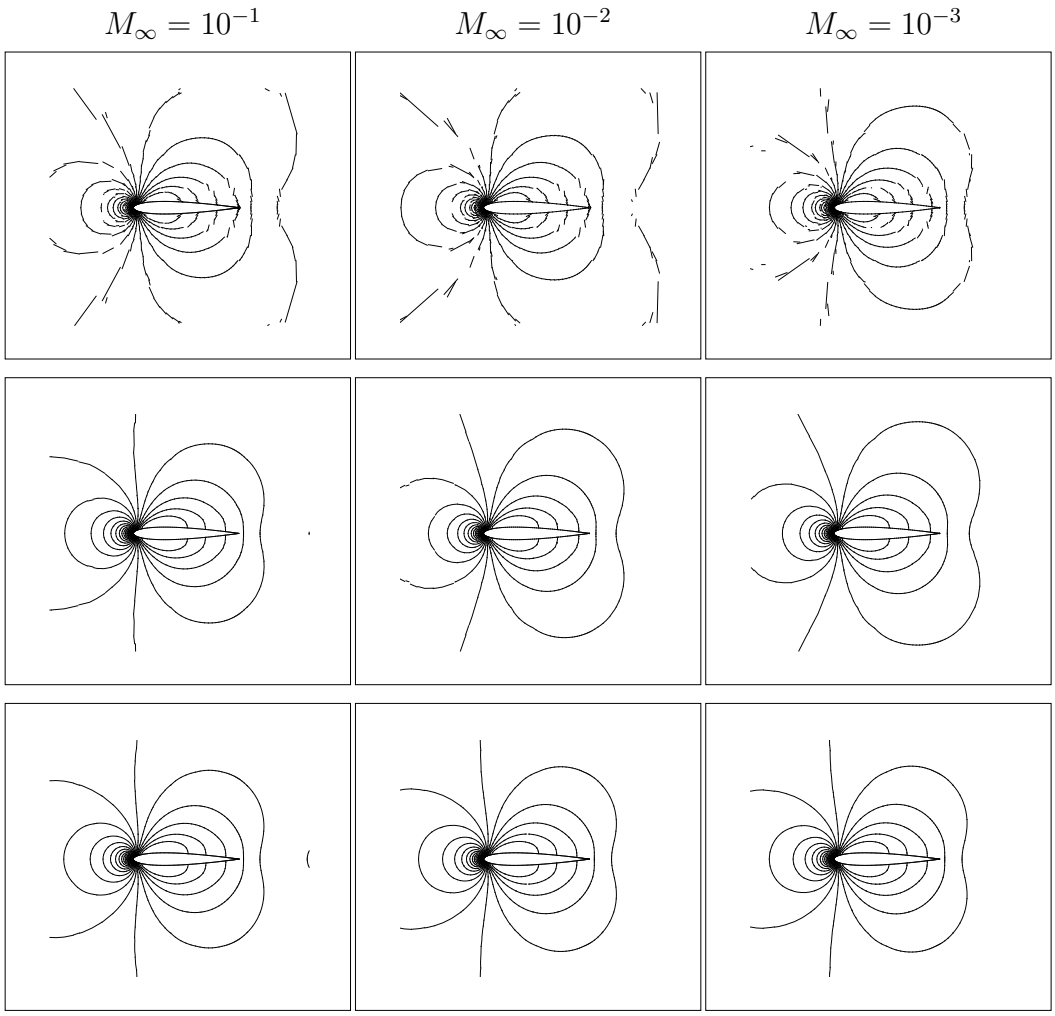


Figure 4.8: NACA0012 test case: contours of normalized pressure with preconditioning for quadrangular grid. $M_\infty = 10^{-1}$ (left column), $M_\infty = 10^{-2}$ (middle column) and $M_\infty = 10^{-3}$ (right column). P_1 (top row), P_2 (middle row) and P_3 (bottom rows) elements.

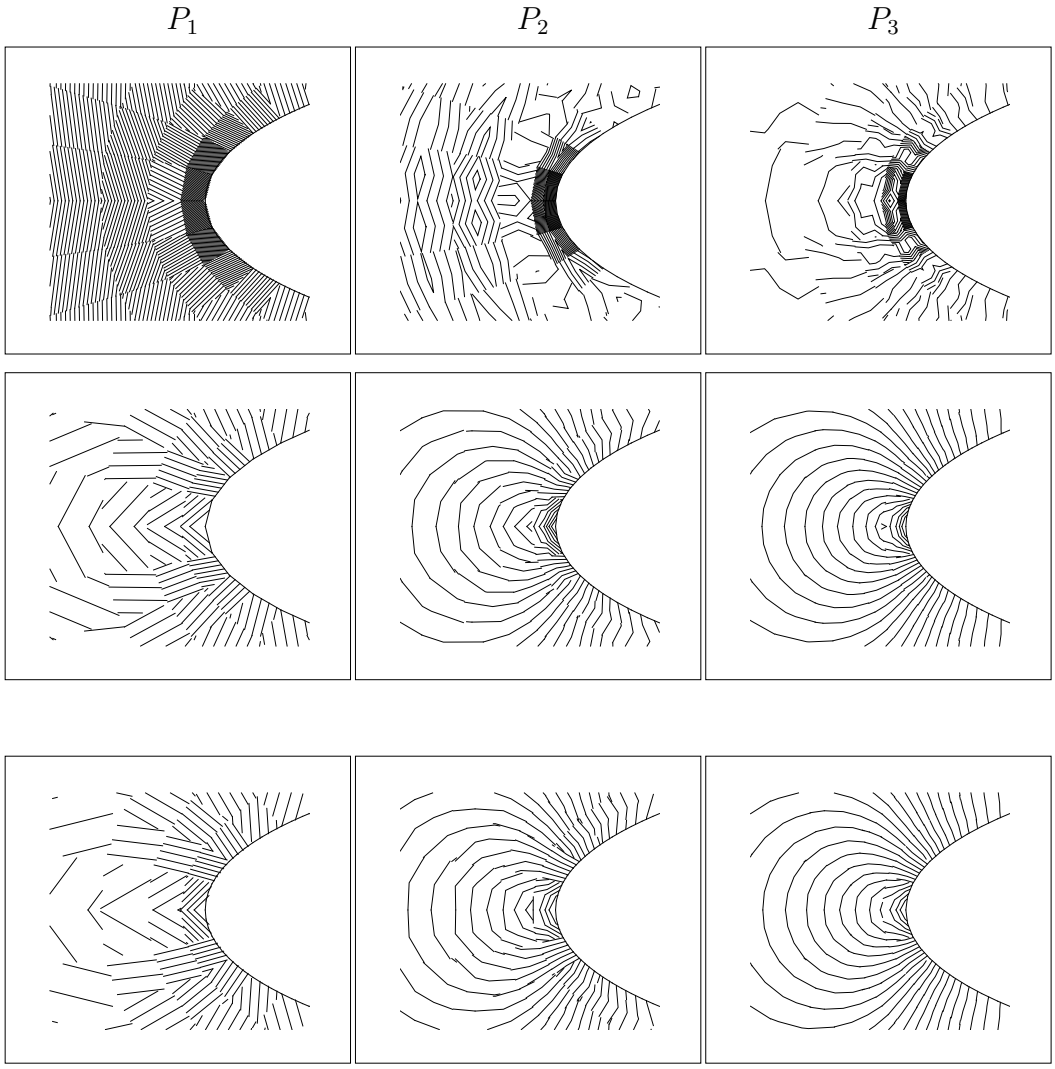


Figure 4.9: NACA0012 test case: contours of normalized pressure near the leading edge for $M_\infty = 10^{-3}$, non-preconditioned (top row), preconditioned (middle row) and non-preconditioned at $M_\infty = 0.4$ (bottom row). Linear, P_1 (left column), quadratic, P_2 (central column), and cubic, P_3 (right column) elements.

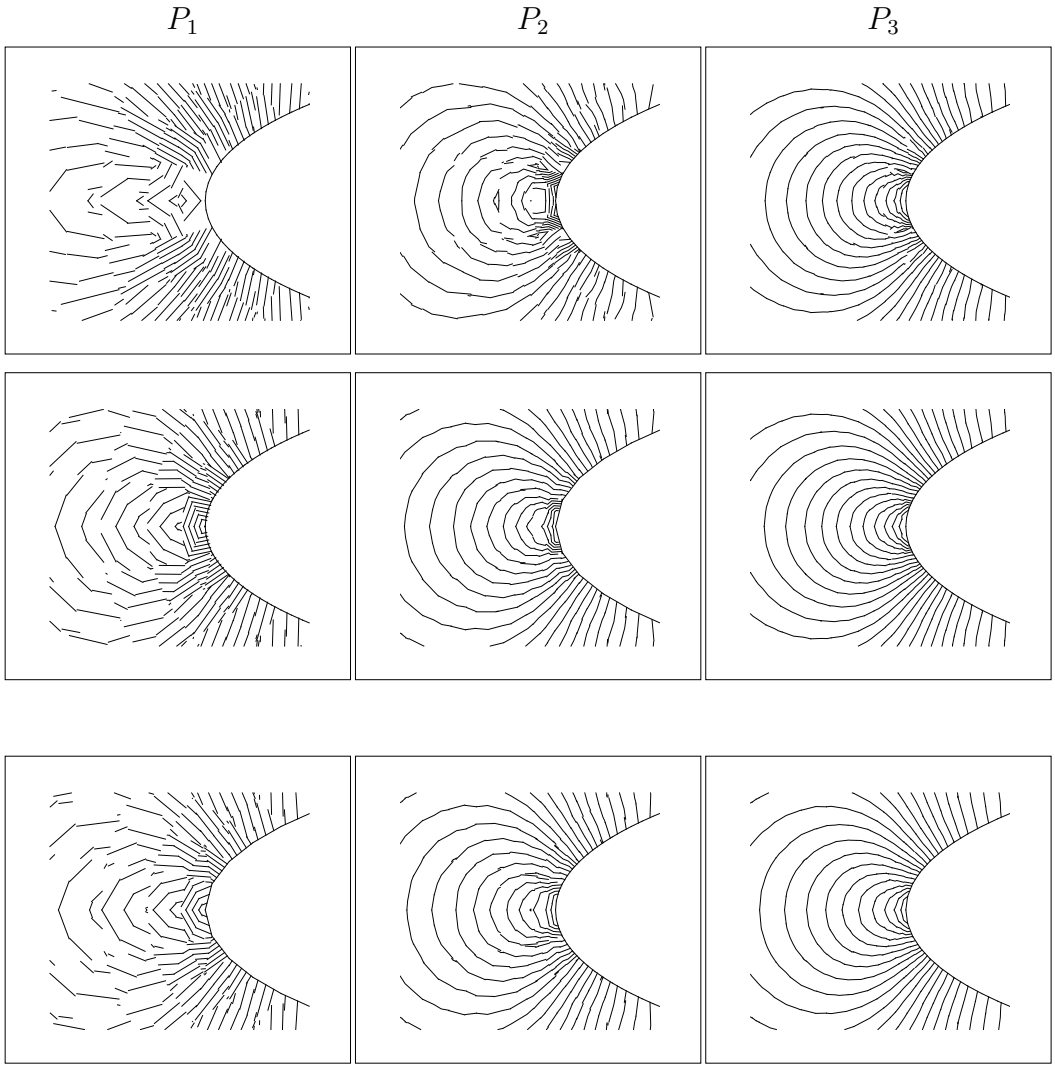


Figure 4.10: NACA0012 test case, triangular grid: contours of normalized pressure near the leading edge for $M_\infty = 10^{-3}$, non-preconditioned (top row), preconditioned (middle row) and non-preconditioned at $M_\infty = 0.4$ (bottom row). Linear, P_1 (left column), quadratic, P_2 (central column) and cubic, P_3 (right column) elements.

4.2.2 Turbulent flat plate

The second test case concerns a turbulent flow horizontally passing over a flat plate. This test case is that reported by Wieghardt [98, 99] and it is one of the validation cases used by the NPARC Alliance to evaluate the accuracy of turbulence models. The Reynolds number based on the plate length is equal to $11.1 \cdot 10^6$. With the aim of reducing the effect of compressibility, the free-stream Mach number, equal to 0.096 in the experiment, is reduced to 0.01 in the computations. The numerical simulations are carried out on a sequence of nested quadrangular grids consisting of 560, 2240, 8960 and 35840 elements respectively, starting from the initial flow field based on the free-stream values, through a sequence of polynomial approximations up to P_4 elements. The meshes are refined near the leading edge of the plate and to better resolve the boundary layer as shown in Figure 4.11. A zero heat flux no slip boundary condition, with ω_w computed following the Menter's projection approach [82], is prescribed from $x = 0$ to $x = 1$ along the lower boundary (flat plate) and symmetry conditions upstream of these points, $x \in [-2, 0]$. At the inflow, the pressure is taken from the flow field, whereas the other variables are prescribed based on the free-stream values, with turbulence intensity, I_t, and turbulent viscosity ratio, $\frac{\mu_t}{\mu}$, set both equal to 10^{-4}. For the upper and downstream boundaries, the pressure is set to the free-stream value. The preconditioning constant ε is set equal to 1. Note that, for some cases which are harder to converge (P_3 elements) it is necessary to increase this value to 3. For this test case there is no presence of stagnation regions unlike in the previous one. Anyway, its solution requires the use of a turbulence model, allowing to gain insight into how well the proposed DG scheme can approximate turbulent boundary layers in the incompressible limit. Finally, we remark that, for each polynomial degree, the initial CFL number of the non-preconditioned computation has been chosen higher than that of the corresponding preconditioned case in order to ensure the convergence of the non-preconditioned solution in

a reasonable number of non-linear iterations.

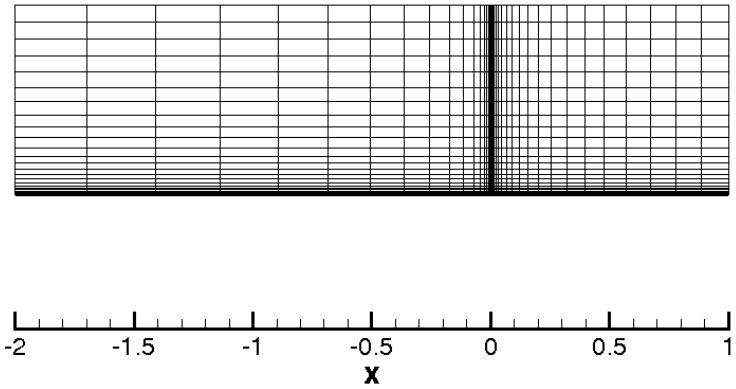


Figure 4.11: Computational grid for the flat plate test case (2240 elements).

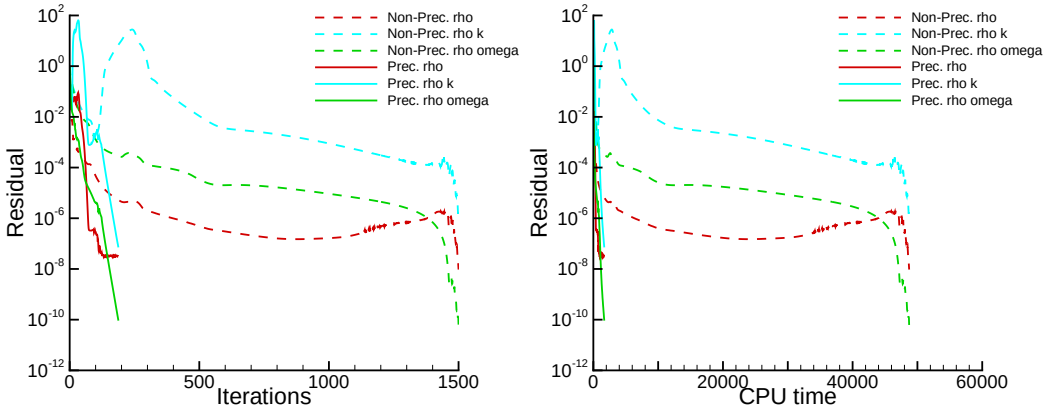
Efficiency of preconditioned schemes

In the flat plate computations, the Krylov process for each linear problem is stopped when the linear residual is reduced by 5 orders of magnitude or after 600 iterations with 1 restart. The Newton iteration is stopped when the L_2-norm of the non-linear residuals is reduced by 10 orders of magnitude. Figure 4.12 compares the history of residuals versus non-linear iterations (left column) and CPU time (right column), with and without full preconditioning. Note that the simulations are carried out using a p-refinement strategy. However, to illustrate more clearly the influence of preconditioning on higher order approximations, the convergence histories are shown separately for the different polynomial degrees without considering the number of non-linear iterations and the CPU overhead required to compute the starting solutions.

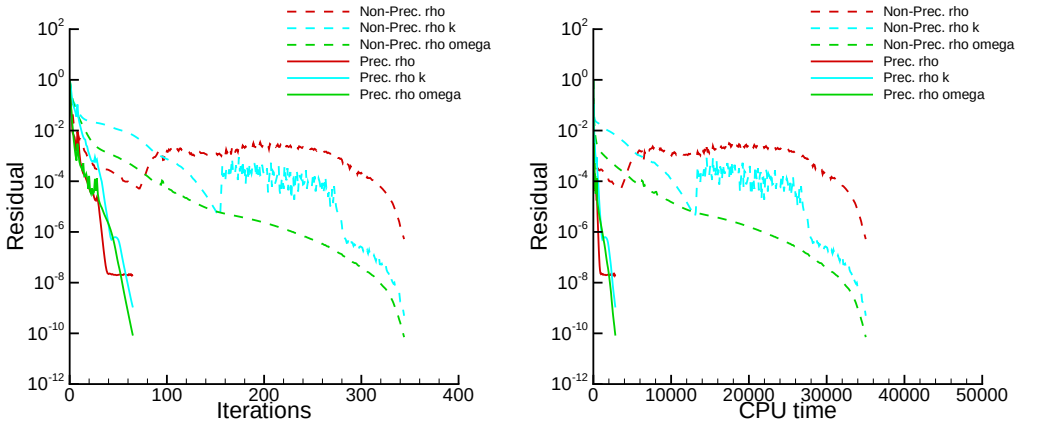
Results refer to computations performed on the mesh with 2240 cells using linear, quadratic and cubic elements. Overall, we see that preconditioning improves the convergence rate and greatly reduces the overhead of CPU time as compared to the non-preconditioned case. This can be already seen for the P_1 and P_2 solutions, where quadratic convergence is achieved in less then 100 non-linear iterations with preconditioning. For P_3 computations, after a drop of 8 orders, residuals increase before reaching the prescribed convergence criterion.

This behaviour of the preconditioned residuals depends on the value of the ε parameter that affects the dissipation introduced by the preconditioning into DG scheme. Specifically, Figure 4.13 shows that an increment of ε from 1 to 3 reduces oscillations in all residual components and thereby improving the convergence rate.

P_1



P_2



P_3

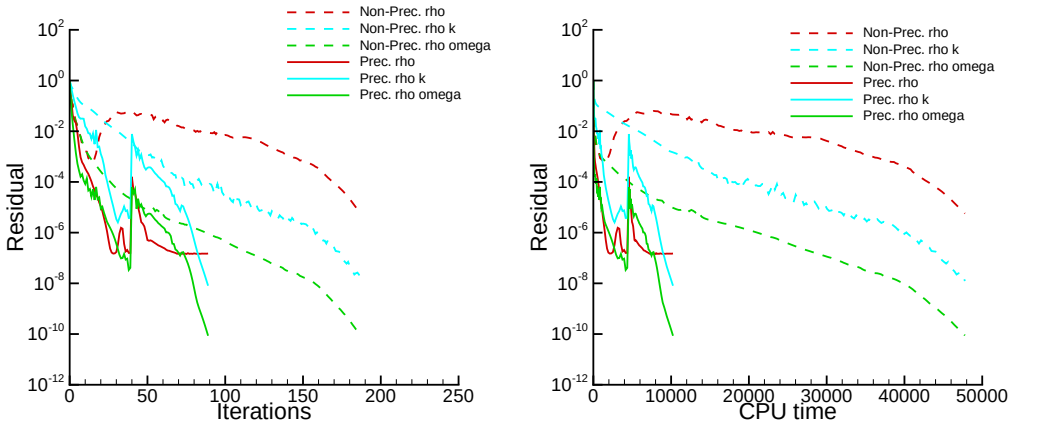


Figure 4.12: Turbulent flat plate: history of L_2-norm of residuals versus number of iterations (left column) and CPU time (right column) at $M_\infty = 10^{-2}$. Linear, P_1 (top row) quadratic, P_2 (middle row) and cubic, P_3 (bottom row) elements.

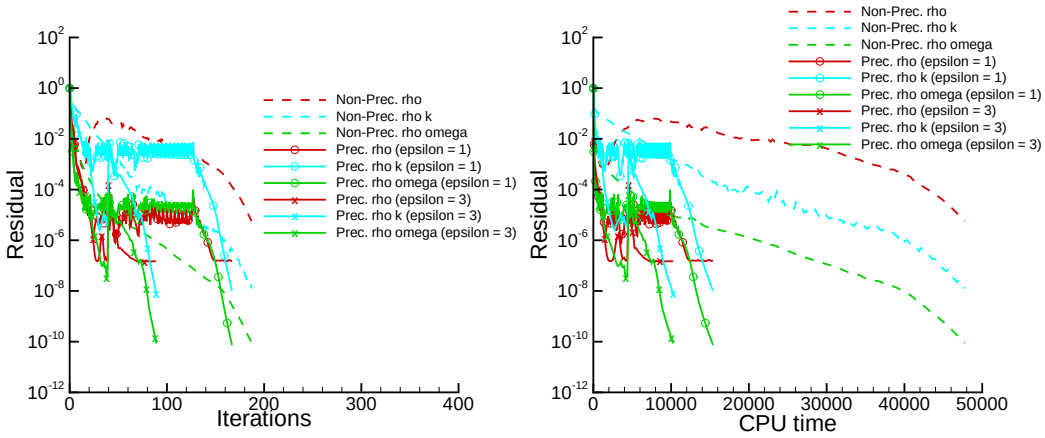


Figure 4.13: Turbulent flat plate, influence of the parameter ε on convergence process: history of L_2-norm of residuals versus number of iterations (left column) and CPU time (right column) at $M_\infty = 10^{-2}$ for P_3 elements.

In Figure 4.14 we present the performance of the GMRES solver with (right column) and without (left column) low Mach number preconditioning for linear, quadratic and cubic elements. The plots on the top row show the number of GMRES iterations (open symbols) and the logarithm of the CFL number (solid symbols), while those on the bottom row show the ratio between the L_2-norms of the last and the first linear residual of the GMRES solver iteration. The quantity along the x-axis is the number of non-linear iterations. In agreement with the results of laminar test case, we can see that, as the CFL number increases, the computations performed without low Mach number preconditioning rapidly use up the maximum number of GMRES iterations (600) without satisfying the required five-order drop of residuals. On the contrary, with preconditioning, for the most part of the non-linear iterations, the required GMRES residual drop is satisfied without reaching the maximum number of linear iterations, thus improving the efficiency of the Krylov solver.

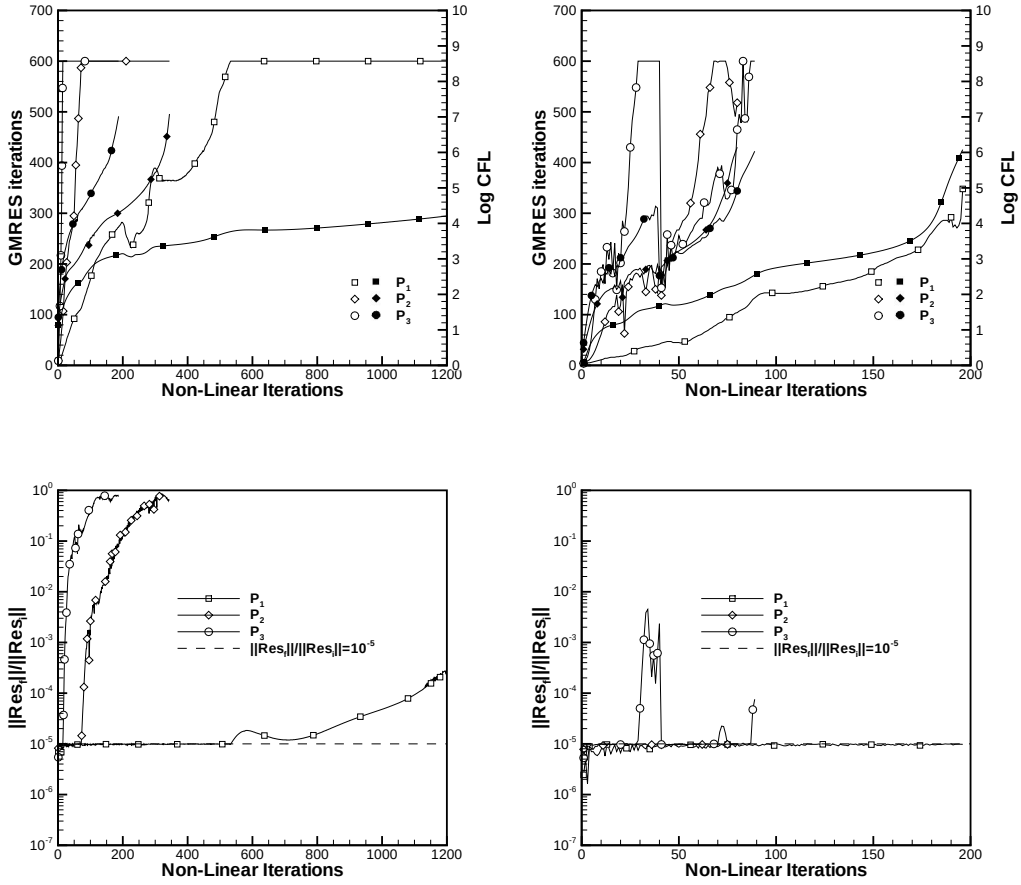


Figure 4.14: Turbulent flat plate: behaviour of the GMRES solver with (right column) and without (left column) low Mach number preconditioning for $M_\infty = 10^{-2}$.

Pressure distribution and skin friction computations

The accuracy capability of the DG scheme can be pointed out qualitatively by the contours of pressure. Figure 4.15 shows the isolines of normalized pressure, at $M_\infty = 0.01$ for P_1, P_2 and P_3 elements without and with preconditioning, respectively. Overall, from the plots, we can appreciate that the preconditioned DG solutions are remarkably more accurate than the corresponding non-preconditioned ones. Furthermore, it is evident that as the polynomial

degree increases, the numerical approximation significantly improves and the difference between preconditioned and non-preconditioned DG solutions reduces.

To analyze more in detail the accuracy of the preconditioned and the non-preconditioned DG schemes in solving the boundary layer over a flat plate, we compare in Figure 4.16 the distribution of the skin friction, c_f, distribution along the wall computed with and without preconditioning, up to fourth-order of accuracy. Firstly we note that, in each case, numerical behaviours are qualitatively similar to the experimental one but with a more dissipative nature, due to the constant eddy viscosity assumption of the $k-\omega$ turbulence model here adopted. Furthermore, the plots show that preconditioning improves the accuracy of the results. In particular, the P_1 solutions are clearly less accurate in the non-preconditioned case than in the preconditioned one. This loss of accuracy is less evident using P_2 and P_3 elements; in fact small differences between preconditioned and non-preconditioned solutions occur only close to the outlet boundary (see the zoom in Figure 4.16). Thus, we can conclude that, the higher the polynomial degree, the lower the difference between preconditioned and non-preconditioned solutions.

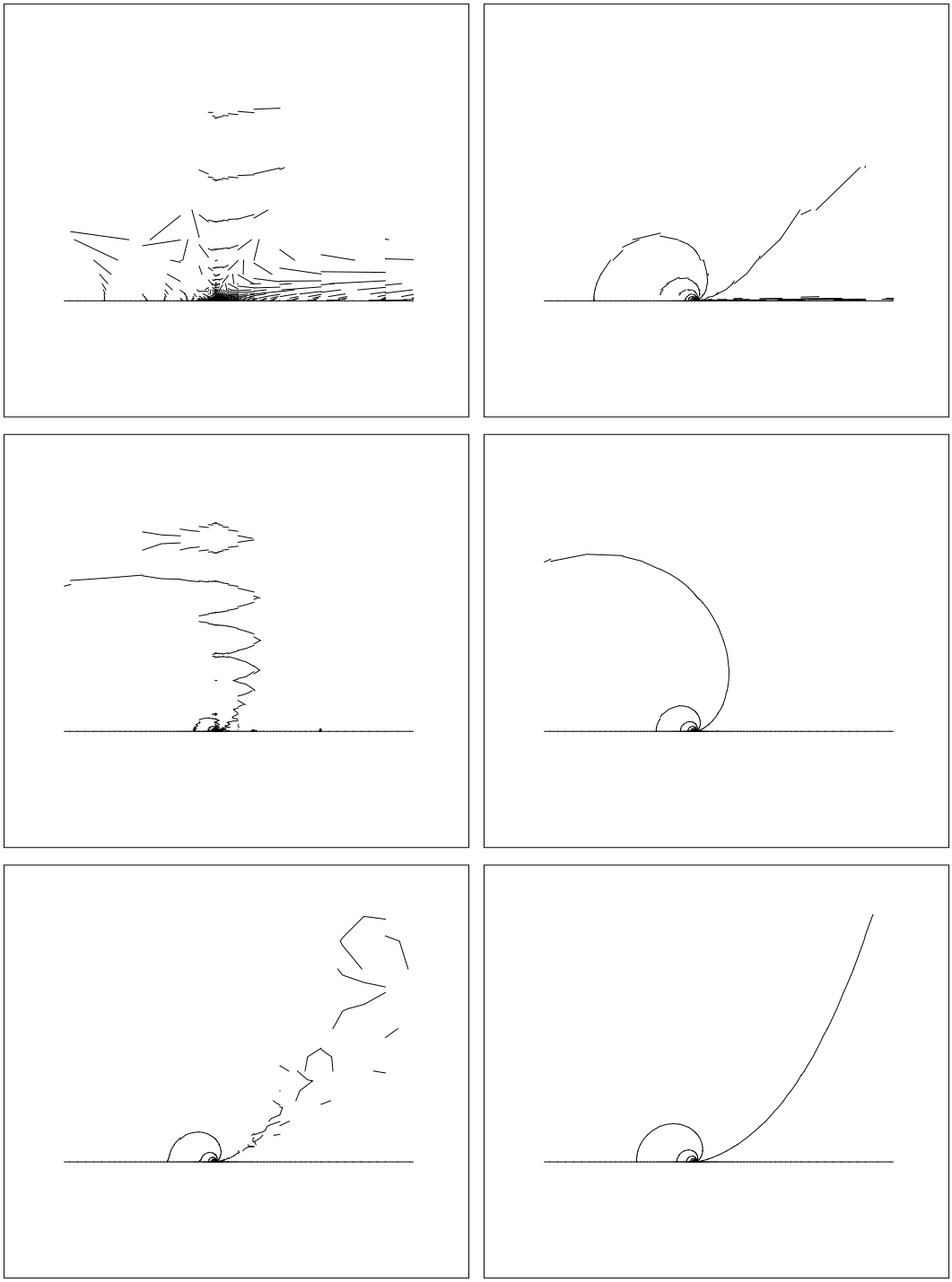


Figure 4.15: Turbulent flat plate: contours of normalized pressure with preconditioning (right column) and without preconditioning (left column) at $M_\infty = 10^{-2}$ for a grid of 2240 elements. P_1 (top row), P_2 (middle row) and P_3 (bottom rows) elements.

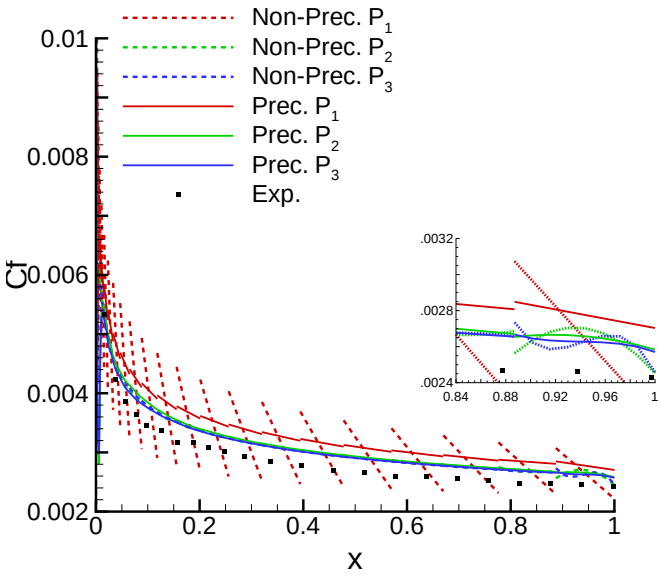


Figure 4.16: Flat plate test case at turbulent conditions: skin friction distribution at wall, with and without preconditioning, for P_1, P_2 and P_3 elements.

Convergence study of the drag coefficient

For this test case, convergence studies of the drag coefficient, C_d, have been performed for high-order discretizations on globally refined meshes. The drag coefficient, C_d, is given by

$$C_d = \frac{2}{\rho_\infty |\mathbf{v}_\infty|^2 l_\infty} \int_S (p\mathbf{n} - \tau\mathbf{n}) \cdot (\cos\alpha, \sin\alpha)^T \, ds, \tag{4.4}$$

where S denotes the surface of the airfoil, l_∞ its chord length, α is the angle of attack, p is the pressure and τ is the total stress tensor, $|\mathbf{v}_\infty|$ and ρ_∞ are the reference velocity and density, respectively [46].

The results are shown in Figure 4.17, where C_d is plotted versus the numbers of degrees of freedom and CPU time for the different polynomial degrees, with and without preconditioning. Overall, we can observe that, in each case preconditioning allows to reduce the computational effort. The right plot of the figure shows that for a given drag value the CPU time required by a preconditioned computation is lower than that for the non-preconditioned one. Concerning the accuracy of the solutions, both the plots put in evidence that

the preconditioned and non-preconditioned drag histories converge towards the same value. Moreover, preconditioning allows to improve the robustness of the scheme. In fact, it permits to compute solutions which do not converge without preconditioning, i.e. P_1 solution on the finest mesh of 35840 cells. The numerical results of both spatial refinement and efficiency are summarized in Tables 4.1, 4.2 and 4.3, where the drag values and CPU times computed on the coarse, medium and fine grid, respectively, for different degrees of freedom are presented.

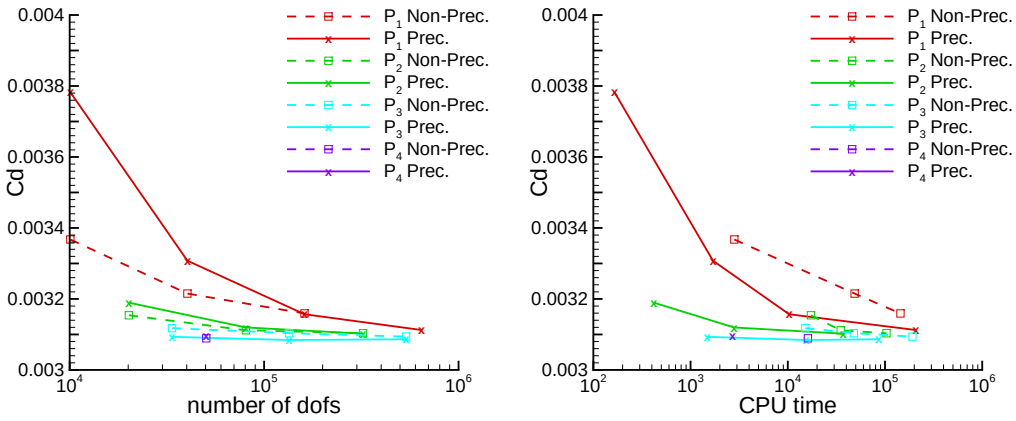


Figure 4.17: Flat plate test case at turbulent conditions: C_d versus degrees of freedom (left column) and versus CPU time (right column).

		Non-Prec.		Prec.	
560 el.	dofs	CPU time (s)	C_d	CPU time (s)	C_d
P_1	10080	2820	0.00337	165	0.00378
P_2	20160	17300	0.00315	420	0.00319
P_3	33600	15150	0.00312	1475	0.00309
P_4	50400	16100	0.00309	2685	0.00309

Table 4.1: Flat plate test case at turbulent conditions: number of degrees of freedom, CPU time and drag coefficient values for P_1-P_4 computations on the coarse mesh.

		Non-Prec.		Prec.	
2240 el.	dofs	CPU time (s)	C_d	CPU time (s)	C_d
P_1	40320	48700	0.00321	1700	0.00331
P_2	80640	35000	0.00311	2800	0.00312
P_3	134400	47800	0.00310	15400	0.00308

Table 4.2: Flat plate test case at turbulent conditions: number of degrees of freedom, CPU time and drag coefficient values for P_1-P_3 computations on the medium mesh.

		Non-Prec.		Prec.	
8960 el.	dofs	CPU time (s)	C_d	CPU time (s)	C_d
P_1	161280	144200	0.00316	10315	0.00316
P_2	322560	103600	0.00310	37150	0.00310
P_3	537600	191620	0.00309	86500	0.00309

Table 4.3: Flat plate test case at turbulent conditions: number of degrees of freedom, CPU time and drag coefficient values for P_1-P_3 computations on the fine mesh.

4.2.3 L1T2 three-element airfoil

In the last test case, we consider a turbulent flow around the L1T2 three-element airfoil at a Reynolds number $Re_\infty = 3.52 \cdot 10^6$ and angle of attack $\alpha = 20.18°$. This problem is aimed at testing the effectiveness of preconditioned and non-preconditioned DG schemes on a two-dimensional low Mach number turbulent flow around a geometrically complex configuration. An original block-structured mesh with 75,840 quadrangular elements has been agglomerated twice resulting in a coarse mesh of 4740 elements. The additional points of the original mesh have been used to define 4740 curved elements, where the mesh edges are treated as quartic polynomials, see Figure 4.18.

At the far-field, a vortex correction boundary condition which specifies density, velocity, turbulent kinetic energy and turbulence dissipation rate at the inflow and pressure at the outflow is used [106]. At the inflow, k and ω are computed based on specified values of turbulence intensity, I_t, and turbulent viscosity ratio, $\frac{\mu_t}{\mu}$, set equal to 10^{-3} and 10^{-2}, respectively. At the wall boundary of the airfoil, we impose a zero heat flux (adiabatic) no-slip boundary condition prescribing ω_w by means of Menter's projection method [82].

The computations have been carried out without and with the full preconditioning approach, at $M_\infty = 0.197$ and at a lower Mach number $M_\infty = 0.01$ in order to reduce the effect of compressibility, using up to a P_3 polynomial approximation. The preconditioning constant ε is set equal to 5 in order to ensure the convergence of numerical solution.

To assess the efficiency of the preconditioned and non-preconditioned DG schemes, convergence histories of residuals versus the number of iterations and versus the CPU time are presented, whereas the accuracy of the numerical solution is analyzed, from a qualitative point of view, by the contour plots of the normalized pressure. Finally, convergence studies of drag, C_d, and lift, C_l, coefficients are presented. Here the C_l coefficient is computed replacing $(\cos\alpha, \sin\alpha)^T$ by $(-\sin\alpha, \cos\alpha)^T$ in Eq. (4.4).

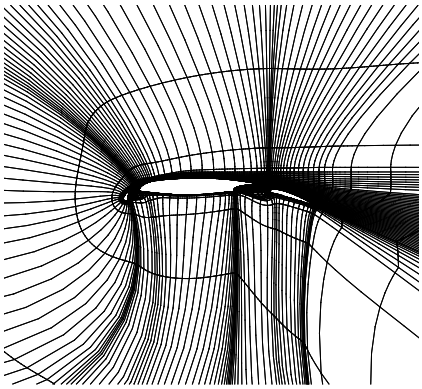


Figure 4.18: Computational grid for the L1T2 test case, 4740 elements.

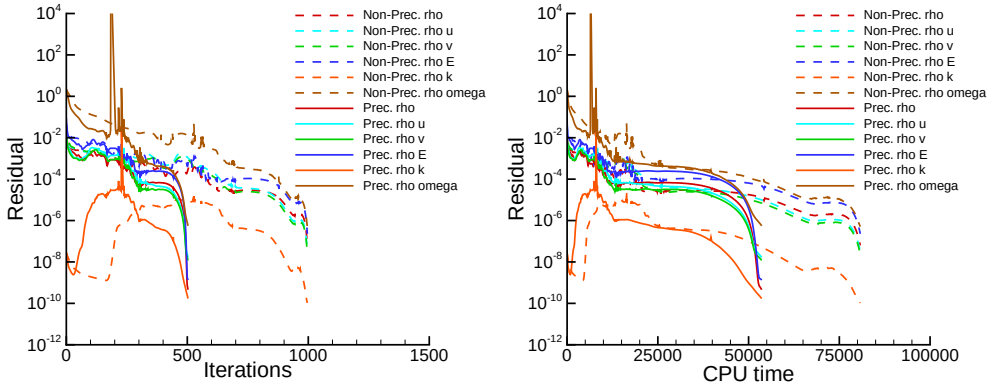
Effects of preconditioning on convergence speed

For this test case the computations have been performed using two different sets of GMRES parameters. On the coarsest mesh the linear system for each non-linear iteration is solved with 240 linear steps and 1 restart; on the refined meshes, the number of linear steps is doubled to 480 with the restart performed after 240 iterations. The linear solver is stopped once the linear residual is reduced by a factor of 10^{-5} or the maximum number of linear iterations is reached. Starting from the initial flow field, based on the free-stream values, the computations have been performed through a sequence of polynomial approximations up to the fourth-order of accuracy.

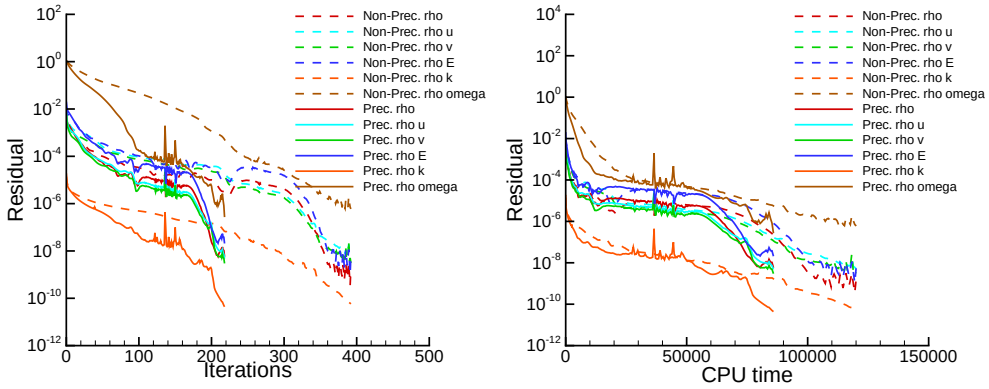
Figure 4.19 compares history of residuals versus non-linear iterations (left column) and CPU time (right column) computed at $M_\infty = 0.197$, with and without full preconditioning. Note that, as for the flat plate, in the plots we compare the number of non-linear iterations and CPU time without considering the computational cost of computing the lower order solutions. The convergence histories are shown for the medium mesh (18960 elements) only as similar convergence histories are obtained on the other ones. The plots show that, for each polynomial degree, preconditioning reduces both the number of non-linear iterations and the computational cost required to satisfy the

convergence criterion despite the Mach number is far from the incompressible limit. In particular, in comparison to the non-preconditioned case, the number of non-linear iterations is almost halved and the overhead of CPU time is reduced approximately by 33%, 28% and 19% for P_1, P_2 and P_3 elements, respectively.

P_1



P_2



P_3

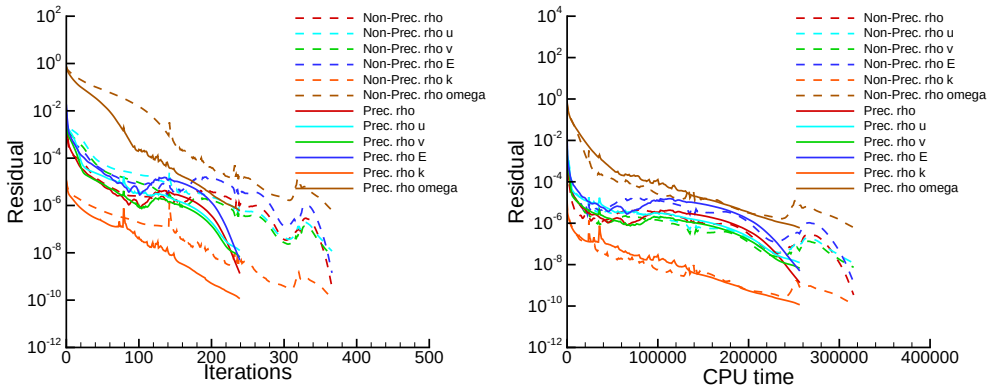


Figure 4.19: L1T2 high-lift configuration: history of L_2-norm of residuals versus number of iterations (left column) and CPU time (right column) on medium mesh (18960 elements) at $M_\infty = 0.197$. Linear, P_1 (top row), quadratic, P_2 (middle row) and cubic, P_3 (bottom row) elements.

Effects of preconditioning on accuracy

Figure 4.20 shows the isolines of normalized pressure computed on the coarsest grid at $M_\infty = 0.197$, using linear, quadratic and cubic elements, without (left column) and with (right column) preconditioning. We can observe that, in all cases, both the non-preconditioned and preconditioned DG solutions yield comparable and very similar contours of pressure. In particular, the P_3 solutions are almost identical whereas only small differences are visible in the P_1 and P_2 solutions. Here, the isolines of the preconditioned solutions are slightly more accurate than those without preconditioning.

In order to investigate in more detail the effectiveness of preconditioning on the accuracy of solution, we performed computations at a lower Mach number to better satisfy the incompressibility condition. Figure 4.21 shows the isolines of normalized pressure computed at $M_\infty = 0.01$, without (left column) and with (right column) preconditioning. Overall, independent of the polynomial degree, the preconditioned isolines are now remarkably more accurate than the corresponding non-preconditioned ones. In particular, we note that even the P_1 isolines are accurately computed with preconditioning, whereas the second-order accurate non-preconditioned contours appear degraded.

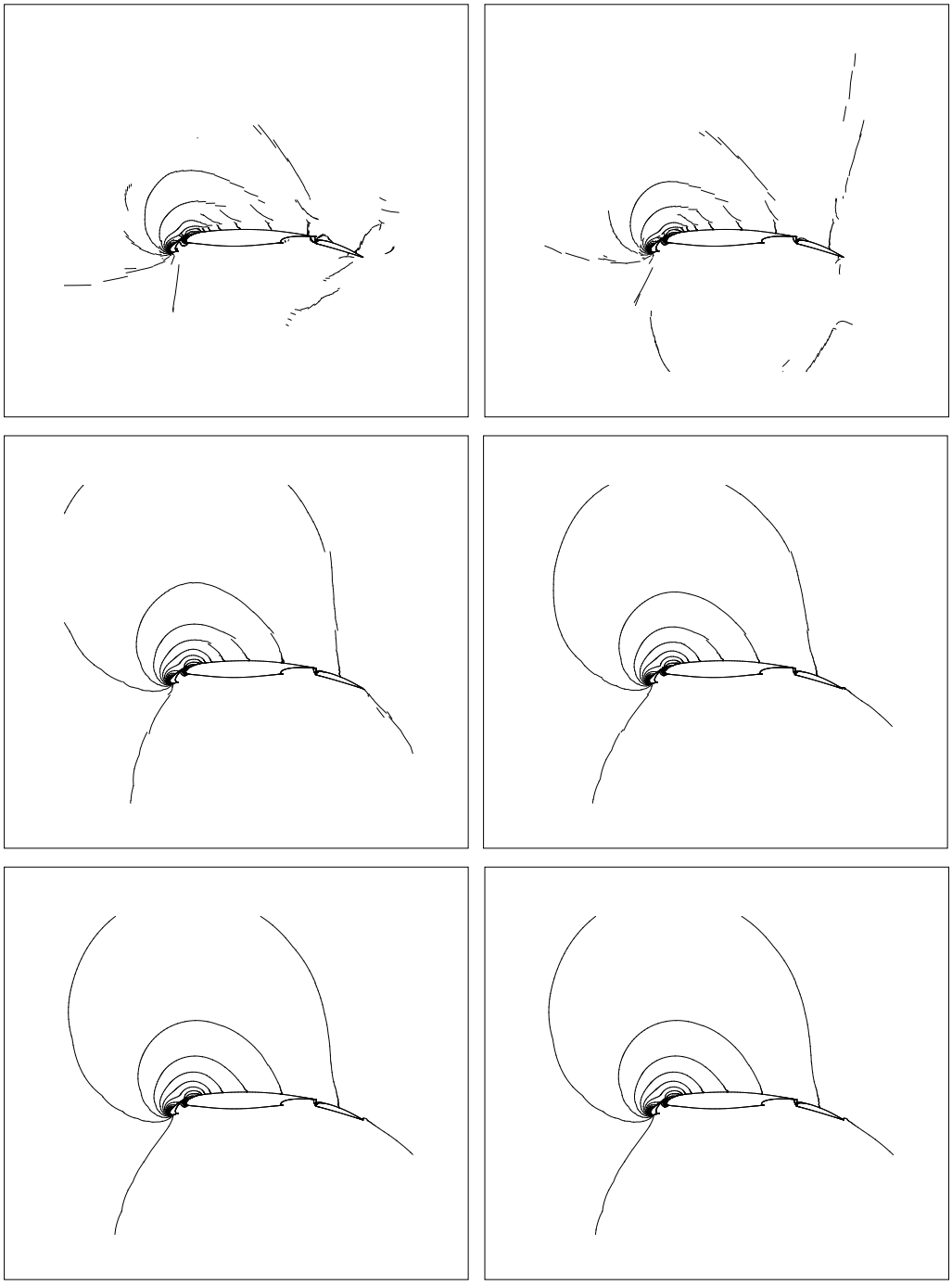


Figure 4.20: L1T2 high-lift configuration: contours of normalized pressure with preconditioning (right column) and without preconditioning (left column) at $M_\infty = 0.197$ on quadrangular grid. P_1 (top row), P_2 (middle row) and P_3 (bottom rows) elements.

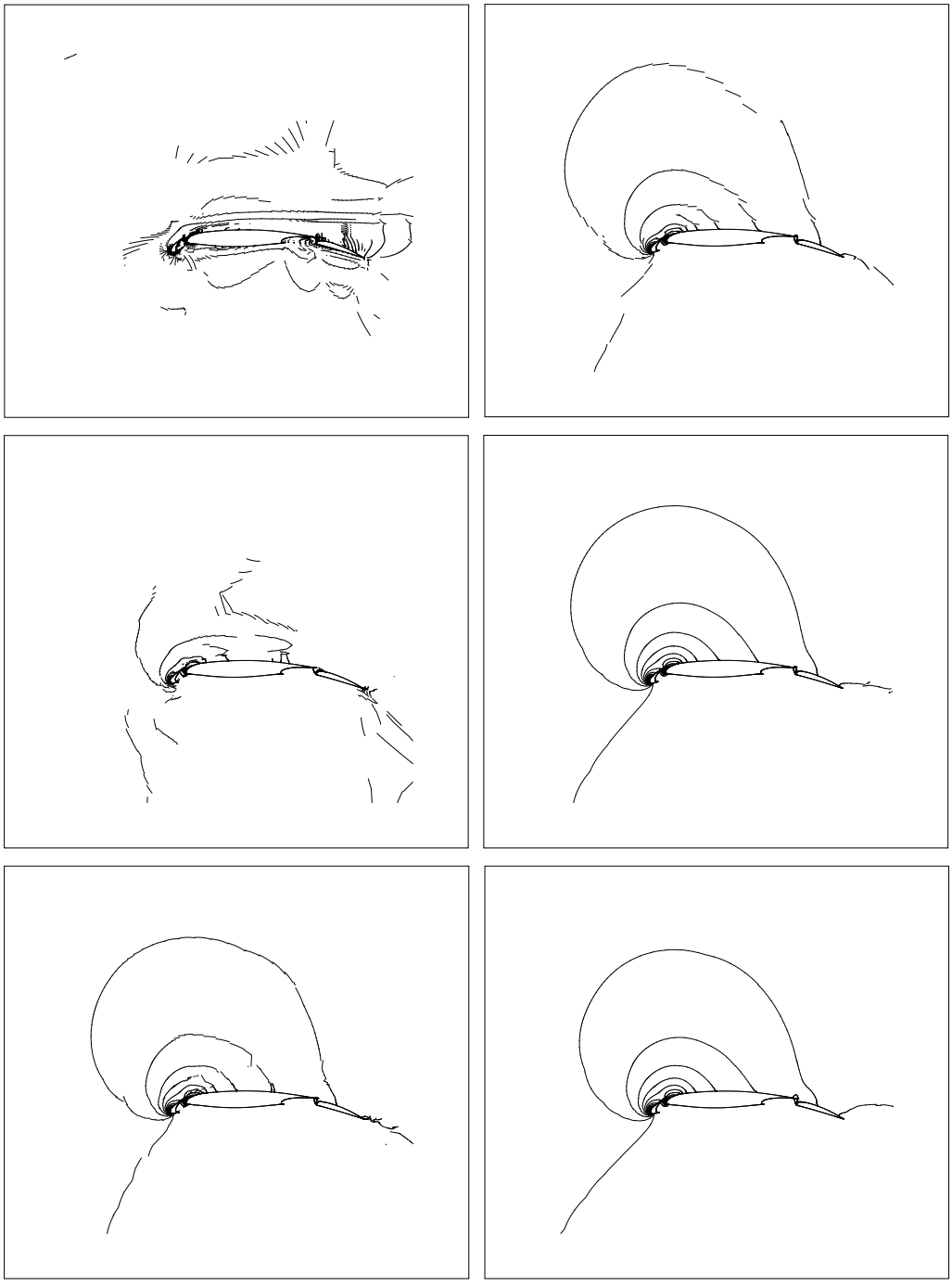


Figure 4.21: L1T2 high-lift configuration: contours of normalized pressure with preconditioning (right column) and without preconditioning (left column) at $M_\infty = 10^{-2}$ on quadrangular grid. P_1 (top row), P_2 (middle row) and P_3 (bottom rows) elements.

Convergence study of drag and lift coefficients

In Figure 4.22 we plot C_d (top row) and C_l (bottom row) values computed at $M_\infty = 0.197$, with and without preconditioning, versus the number of degrees of freedom (left column) and CPU time (right column). As convergence criterion for achieving the steady-state force coefficients, a non-linear residual reduction of 8 orders of magnitude from the initial residual, based on free-stream conditions, is considered. The graphs show results for linear, quadratic and cubic elements on coarse and medium meshes, and for linear and quadratic elements on the finest mesh. Although being far from the grid converged the results in the plots show that, in agreement with the results obtained for the turbulent flat plate, the force coefficients converge towards similar values with and without preconditioning. Furthermore, preconditioned computations are more efficient than the non-preconditioned ones where the savings in CPU time become larger as the number of degrees of freedom increases. These results are summarized in Tables 4.4, 4.5 and 4.6, where the drag values and CPU times computed on the coarse, medium and fine grids, respectively, for different degrees of freedom, are presented.

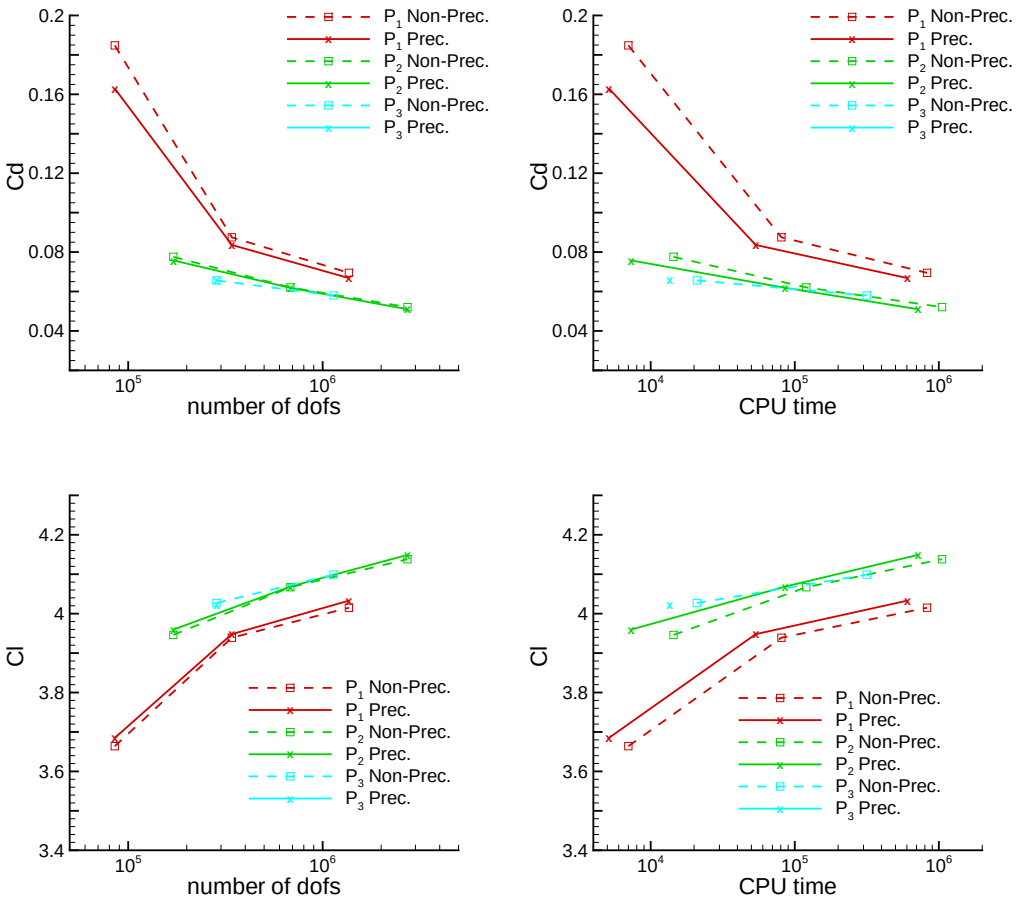


Figure 4.22: L1T2 high-lift configuration: C_d (top row) and C_l (bottom row) versus degrees of freedom (left column) and versus CPU time (right column) at $M_\infty = 0.197$.

		Non-Prec.			Prec.		
4740 el.	dofs	CPU (s)	C_d	C_l	CPU (s)	C_d	C_l
P_1	85320	7000	3.664	0.01848	5130	3.685	0.16293
P_2	170640	14400	3.946	0.07759	7300	3.989	0.07577
P_3	284400	20900	4.027	0.06562	13600	4.023	0.06602

Table 4.4: L1T2: number of degrees of freedom, CPU times and force coefficients values of P_1-P_3 computations at $M_\infty = 0.197$ on the coarse mesh.

		Non-Prec.			Prec.		
18960 el.	dofs	CPU (s)	C_d	C_l	CPU (s)	C_d	C_l
P_1	341200	80800	3.939	0.08748	53600	3.948	0.08361
P_2	682560	120100	4.067	0.06207	85900	4.069	0.06186
P_3	1137600	315800	4.099	0.05798	256400	4.094	0.05833

Table 4.5: L1T2: number of degrees of freedom, CPU times and force coefficients values of P_1-P_3 computations at $M_\infty = 0.197$ on the medium mesh.

		Non-Prec.			Prec.		
75840 el.	dofs	CPU (s)	C_d	C_l	CPU (s)	C_d	C_l
P_1	1365120	828000	4.015	0.06947	604650	4.033	0.06680
P_2	2730240	1052000	4.138	0.05207	717000	4.149	0.05102

Table 4.6: L1T2: number of degrees of freedom, CPU times and force coefficients values of P_1-P_2 computations at $M_\infty = 0.197$ on the fine mesh.

Chapter 5

Rescaling of the RANS $k-\omega$ equations

In this chapter we present several rescaling techniques applied to the RANS $k-\omega$ equations in combination with a DG spatial discretization. With the aim to develop a more efficient and robust high-order flow solver, two kinds of scalings are considered. Firstly, the governing equations are rescaled based on metric terms such as the inverse cell area or the inverse mass matrix and/or based on turbulent quantities. Secondly, a so-called auto-scaling approach is implemented, which rescales the linear system ($\mathbf{Ax} = \mathbf{b}$) arising in an implicit solution algorithm. Note that, all the remarks and techniques presented in this chapter can be generalized and adapted to other numerical schemes and turbulence models.

5.1 Scaling problems

The entries of the global residual vector may be highly unbalanced because of poor scaling of variables and equations. Scaling problems can be detrimental for the efficiency of implicit solvers in which the linear systems are solved inexactly and the time step evolution depends on the convergence of the residual components.

A first reason for the disparity between residual components is concerned with geometric features and it is specially related to the computational mesh and coordinate transformation of elements from reference to physical space. Since the equations have an element-by-element scaling of the metric Jacobian, the residual at cells far from the body, where metric Jacobian is large, is generally severals orders of magnitude higher than the residual at cells close to the wall boundary, where the metric Jacobian is small due to the finer mesh targeted to resolve the boundary layer.

Secondly, the residuals are unbalanced due to the fact that mean-flow and turbulent variables differ by several orders of magnitude. For example, when the $k-\omega$ model [77, 81] is used, the turbulent variable ω theoretically could range up to infinity (value at wall), while the mean-flow variables typically do not exceed two. More in detail, mean-flow variables are typically of the order of 1, k is in the order of $0.001 \div 0.1$, whilst ω takes sufficiently high values that approximate infinity, such as $10^6 \div 10^9$. Since in the present work the turbulence model is implemented in terms of $\widetilde{\omega} = ln\ \omega$ instead of ω, a value of $\widetilde{\omega}$ equal to 15 or 20 can represent its upper bound in numerical simulations. A similar disparity affects the residual when other turbulence models are used, such as the Spalart-Allmaras model [67], where the turbulent viscosity variable, $\widetilde{\nu}$, ranges up to roughly 1000. As a result, turbulent residuals can be orders of magnitude larger than that of the mean-flow equations. Furthermore, also the entries in each Jacobian block of the linear system matrix are affected by the lack of proper scaling due to the disparity between mean-flow and

turbulent variables. In particular, large differences arise in entries resulting from the derivative of mean-flow quantities with respect to the turbulent ones. Lastly, local changes in residuals potentially can occur when highly non-linear components are present in the physical model or the discretization, such as in several turbulence models [67, 75] or shock capturing schemes [44, 45].

Since the linear systems arising in the implicit solver are solved inexactly, i.e. by means of the GMRES algorithm [36] with a stopping criterion based on the maximum number of Krylov steps or a linear residual reduction, it may be that only the residual at cells far from the body is reduced by the approximate linear solver, while the residual at cells close to wall boundaries can potentially increase. Similarly, only the residual of the turbulence model can be reduced while the residual of the mean-flow equations may increase. Now, if the linear residual in one of the mean-flow equations is not sufficiently decreased, or even increased, this could cause unphysical values in the next non-linear solution iteration (e.g. density/pressure < 0) and thus a break-down of the non-linear solution process. Additionally, addressing scaling problems might allow to use larger time steps and higher tolerances of the linear solver, i.e. a more inexact solution, resulting in a saving of computational time.

Finally, when the CFL strategy evolution is based on the Switched Evolution Relaxation (SER) or Residual Difference Method (RDM) algorithms [34, 35], the sequence of CFL numbers selected is not only determined by the choice of the control parameters, but also on the norm of the residual. Therefore a disparity between residual components can adversely affect the time step size. In particular, following the SER strategy, the CFL number at the n-th iteration is defined as follows

$$CFL_n = min\left(CFL_0\left(\frac{\|\mathbf{R}_0\|_2}{\|\mathbf{R}_n\|_2}\right)^{\alpha}, CFL_{max}\right) \tag{5.1}$$

where $\|\mathbf{R}_0\|_2$ and $\|\mathbf{R}_0\|_n$ are the L_2-norms of residual at the initial and current iteration step, respectively. CFL_0 is the initial CFL number, while CFL_{max} and α are two control parameters. For turbulent flows, CFL_n will be strongly

influenced by the ω-equation. As a result, convergence problems may occur in the other equations due to an inappropriate choice of the CFL number. Finally, we remark that, even if the dependence of the CFL number from a single equation is released, i.e. CFL is not related/is weakly related to a particular equation (like e.g. in laminar case), the L_2-norm computation still remains influenced by the metric scaling due to the grid dependence of the residuals.

5.2 Rescaling techniques

Problems caused by a lack of scaling in the metric and the equations can be addressed by means of rescaling techniques. A generic scaling can be applied to the linear system $\mathbf{Ax} = \mathbf{b}$ of Eq. (3.9) as row scaling as follows

$$\mathbf{SAx} = \mathbf{Sb}, \tag{5.2}$$

where $\mathbf{S}$ is a vector, or equivalently a (block) diagonal matrix that scales the residual and the matrix $\mathbf{A}$ in the linear system.

Two kinds of rescaling approaches are considered: inherent scaling (*a priori* or *ab initio* scaling) and auto-scaling. The former aims to overcome permanent scaling problems such as grid dependence and inherent differences between turbulence and mean-flow variables and has an effect on the time step computation. Basically, it is equivalent to rewrite the equations in a different way. The latter is implemented to avoid that by solving the linear system inaccurately, the residuals of mean-flow equations increase resulting in negative pressures or densities and is calculated on an iteration-by-iteration basis.

5.2.1 *A priori* scalings

To eliminate the influence of local element sizes, the linear system needs to be scaled by some metric of the grid closely related to the Jacobian of the coordinate transformation, such as the cell area (or volume), J_K, of each element K,

evaluated by integrating the determinant of the mapping over the reference element $\hat{K}$ [110]. The resulting scaling vector $\mathbf{S}_{J^{-1}}$ of size $m \times N_{dof}^K \times N_K$, with N_K number of elements and N_{dof}^K number of dofs in the generic element K for each of the m conservative variables, is given by

$$\mathbf{S}_{J^{-1}} = \{\mathbf{J}_1{}^{-1}, \mathbf{J}_2{}^{-1}, ..., \mathbf{J}_{N_K}{}^{-1}\}^T,$$

where $\mathbf{J}_K{}^{-1}$ defined as follows

$$\mathbf{J}_K{}^{-1} = \{J_K{}^{-1}, J_K{}^{-1}, ..., J_K{}^{-1}\}_{m \times N_{dof}^K}.$$

The cell area J_K is a constant quantity of the element K, therefore the scaling factor $\frac{1}{J_K}$ is the same for all of the $m \times N_{dof}^K$ dofs of the element K. Notwithstanding, in DG methods, it's common practice to deal with curved elements by employing higher order polynomial mappings of the reference element $\hat{K}$ onto the element K in the physical space. Consequently, the Jacobian of the mapping is not generally constant within the element. To take into account the variation of the metric Jacobian within the cell, a more general metric scaling approach consists in replacing the vector $\mathbf{S}_{J^{-1}}$ by the inverse of the mass matrix $\mathbf{M}$, with

$$M_{ij} = \int_\Omega v_i v_j \, d\mathbf{x}, \quad i, j = 1, ..., m \times N_{dof}^K \times N_K.$$

Each diagonal block of $\mathbf{M}^{-1}$, which represents the inverse of the element mass matrix $\mathbf{M}_K$, with $M_{Kij} = \int_K v_i v_j \, d\mathbf{x}$, $i, j = 1, ..., m \times N_{dof}^K$, better approximates the Jacobian within the cell K. The application of either of these scalings should yield a reduction of the ill-conditioning of the linear system due to the grid dependence. However, disparity between residual components may still persist for turbulent computations due to the difference among the magnitude of mean-flow and turbulent variables. For the purpose of limiting this source of unbalance, another scaling, impacting on turbulence model equations only, needs to be applied. More in detail, the turbulence model equations can be scaled by the maximum values of the transported turbulent variables k_{max} and $\widetilde{\omega}_{max}$. The resulting scaling vector is given by

$$\mathbf{S}_{turb} = \{\mathbf{J}_{k\widetilde{\omega}}{}^{-1}, \mathbf{J}_{k\widetilde{\omega}}{}^{-1}, ..., \mathbf{J}_{k\widetilde{\omega}}{}^{-1}\}_{N_{dof}^{K} \times N_K}{}^{T},$$

with

$$\mathbf{J}_{k\widetilde{\omega}}{}^{-1} = \{1, 1, 1, 1, \hat{k}^{-1}, \hat{\widetilde{\omega}}^{-1}\}_m.$$

Here $\hat{k}$ and $\hat{\widetilde{\omega}}$ are approximations of k_{max} and $\widetilde{\omega}_{max}$, respectively. Although the maximum values of k and $\widetilde{\omega}$ depend on the flow solution, they must be specified *a priori*. In particular, $\hat{k}$ could be estimated from semi-empirical correlations for simple flows such as the flat plate. A more general strategy in the framework of DG methods would be the computation of the maximum k value, for a given polynomial degree p, from the discrete solution of polynomial degree $p-1$. Anyway, in our numerical experiments it was found that $\hat{k} \approx 0.03$ works well for aerodynamic computations of flows around airfoils. Concerning $\hat{\widetilde{\omega}}$, DG computations showed that a value between $15 \div 20$ can be specified without significant changes in convergence history and numerical results.

Once metric and turbulent scalings vectors are built, they can be combined and applied to the linear system of Eq. (3.9) as follows

$$\left(\mathbf{S_{J^{-1}} S_{turb}}\right) \mathbf{A x} = \left(\mathbf{S_{J^{-1}} S_{turb}}\right) \mathbf{b}$$

or

$$\left(\mathbf{M^{-1} S_{turb}}\right) \mathbf{A x} = \left(\mathbf{M^{-1} S_{turb}}\right) \mathbf{b},$$

resulting in a new rescaled system

$$\mathbf{A_S x} = \mathbf{b_S}. \tag{5.3}$$

Hence, the new CFL number, and the local time-steps, are computed based on the L_2-norm of the scaled global vector of residuals $\mathbf{b}_S$.

5.2.2 Auto-scaling

Despite the use of *a priori* scaling techniques, disparities still persist between the entries of $\mathbf{b}_s$ when solving the linear system. To reduce these unbalances,

a further scaling, called auto-scaling, $\mathbf{S}_{auto}$, is applied to the rescaled system of Eq. (5.3) as follows,

$$\mathbf{S_{auto}}\mathbf{A_S}\mathbf{x} = \mathbf{S_{auto}}\mathbf{b_S}. \tag{5.4}$$

The auto-scaling $\mathbf{S}_{auto}$ is calculated, at each non-linear iteration, in order to keep the maximum difference between residual norms to within a specified order of magnitude (typically one order). Specifically, we build $\mathbf{S}_{auto}$ based on a vector $\mathbf{R}_s$, which depends on the scaled residual vector $\mathbf{b}_s$. The components of $\mathbf{S}_{auto}$ are computed by imposing the constraint that the entries of $\mathbf{S}_{auto}\mathbf{R}_s$ are within a prescribed tolerance. $\mathbf{R}_s$ is the vector with components given by the L_2-norms of residuals computed for each degree of freedom of each equation as follows,

$$R_{Sj} = \sqrt{\frac{\sum_{i=1}^{N_K} {b_{sj,i}}^2}{N_K}}, \qquad j = 1, ..., m \times N_{dofs}^K,$$

with $b_{sj,i}$, entry of the scaled residual vector $\mathbf{b}_s$, associated to the j-th dof of the element i. We recall that N_{dofs}^K is the number of degrees of freedom for each of the m governing equations and N_K is the total number of elements. Note that, in order to build $\mathbf{R}_s$ and thus $\mathbf{S}_{auto}$ others strategies have been implemented and tested, although without substantial changes in results.

Finally, we remark that auto-scaling has an influence on the linear solver only, since the time step computation is performed before the auto-scaling is applied to the linear system.

5.3 Numerical Results

In this section two aerodynamic test cases are presented in order to asses the effectiveness of the rescaling algorithms in the framework of DG schemes: an inviscid transonic flow around the NACA0012 airfoil and a turbulent transonic flow around the RAE2822 airfoil. The test cases are both characterized by the presence of shocks. Hence a stabilization technique, as described in Section 3.1.1 is used to stabilize the discretization near the shocks. Comparisons between non-scaled and scaled results obtained using cell metrics, turbulent quantities and auto-scaling are presented.

The computations are performed using various degrees of polynomial approximations for the purpose of demonstrating the effect of the different rescaling techniques in improving the efficiency of a high-order flow solver. Computations are carried out for fixed GMRES parameters (number of Krylov vectors, restarts and relative tolerance to stop iterative solution), with the convergence of the solution process presented in terms of histories of the lift coefficient, C_l, and of the L_2-norm of residuals versus number of iterations and CPU time (seconds), respectively.

5.3.1 Inviscid flow around a NACA0012 airfoil with shock

In the first test case, we consider an inviscid flow around a NACA0012 airfoil. The computational domain consists of 1600 quadrilateral elements, with curved boundaries approximated by piecewise quadratic polynomials, see Figure 5.1. At the far-field boundary we specify a Mach number $M_\infty = 0.8$ at an angle of attack $\alpha = 1.25°$; on the wall, we set a slip-wall boundary condition which imposes $\mathbf{v} \cdot \mathbf{n} = 0$ to guarantee no flow normal to the surface. For this test case we compute the flow solutions up to the fifth-order of accuracy using tensor product Lagrange polynomials. In each non-linear solution step a linear system is solved with the restarted and block-ILU-preconditioned GMRES method. A maximum of 180, 240 and 300 Krylov vectors has been specified

for quadratic, cubic and quartic elements respectively, with a restart of the algorithm performed after 120 iterations. The linear solver is stopped once the maximum number of linear iterations is reached or the initial linear residual is reduced by a factor of 10^{-6} or 10^{-1}, respectively.

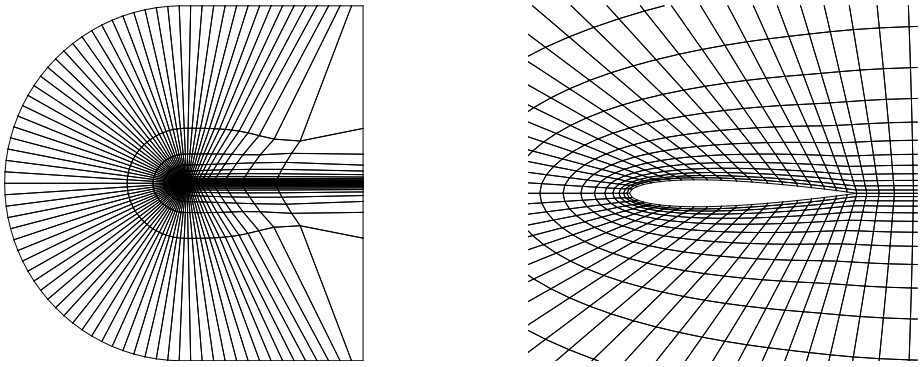


Figure 5.1: Computational grid, NACA0012 test case.

First, we investigate the influence of various scalings on the convergence rate of the non-linear solution algorithm. To this end, Figure 5.2 compares the C_l convergence histories in terms of number of non-linear iterations and CPU time using the non-scaled (left) and the scaled (right) DG algorithms for a linear tolerance of 10^{-6}. In particular, we consider the use of the metric scaling, the inverse cell area $\mathbf{S}_{J^{-1}}$ or the inverse mass matrix $\mathbf{M}^{-1}$, with and without auto-scaling. The plots show that, in all cases, the non-scaled and scaled algorithms yield comparable lift behaviours and at convergence, the lift value is the same for both the schemes. This is due to the fact that scalings impact on efficiency, without altering the accuracy of the fully converged solution. In fact, the scaled solutions converge in a smaller number of non-linear iterations than the non-scaled ones, with the benefit of reducing the computational effort. Furthermore, we can appreciate that there is no substantial difference in the P_2 and P_3 results obtained using, either the inverse cell area or the inverse

mass matrix as scaling factor, whereas only small differences appear for P_4 solutions. Finally, Figure 5.2 shows that auto-scaling has almost no effect on the accuracy and efficiency of the $\mathbf{J}^{-1}$ or $\mathbf{M}^{-1}$ rescaled solutions.

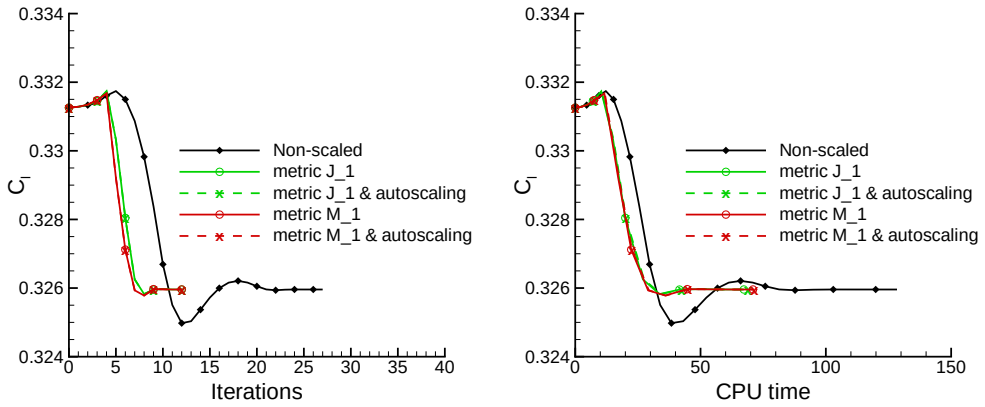
To evaluate the impact of the scalings on the inexact solution of the linear system more in depth, Figure 5.3 shows the convergence histories of the lift coefficients obtained considering a linear tolerance of 10^{-1}. Also in this case the scaled solutions converge faster than the non-scaled ones. In particular, the rescaling shows remarkably higher efficiency for third- and fifth-order accurate solutions with just slight improvements in the P_3 computations. Furthermore, we can see that the metric scaling based on the inverse cell area is more efficient than the inverse mass matrix approach. The saving of CPU time, which is negligible for P_2 computations, increases with the polynomial degree. Unlike previous results, the combination of auto-scaling and metric scaling is now shown to have some influence on the convergence histories, although its contribution to the improvement of the performance is less significant. Finally, the comparison between the corresponding C_l versus CPU time behaviours of Figure 5.3 and Figure 5.2 shows that a linear tolerance of 10^{-1} improves the efficiency of the implicit solver, independently of the metric scaling.

These results point out that only the metric scaling, in particular if based on the cell area, significantly improves the convergence rate, whilst auto-scaling poorly influences the performance of the scheme. However, as all the scalings, except of the auto-scaling, impact on both the time step evolution and the solution of the linear system, further investigations have been carried out to gain more insight into how scalings improve the performance of the scheme. Figure 5.4 shows histories of the lift coefficient obtained by applying the inverse cell area scaling to the time step computation and to the linear system, separately. All plots suggest that rescaling mainly contributes to the computational efficiency through the time step, whereas its influence through the linear system is almost negligible.

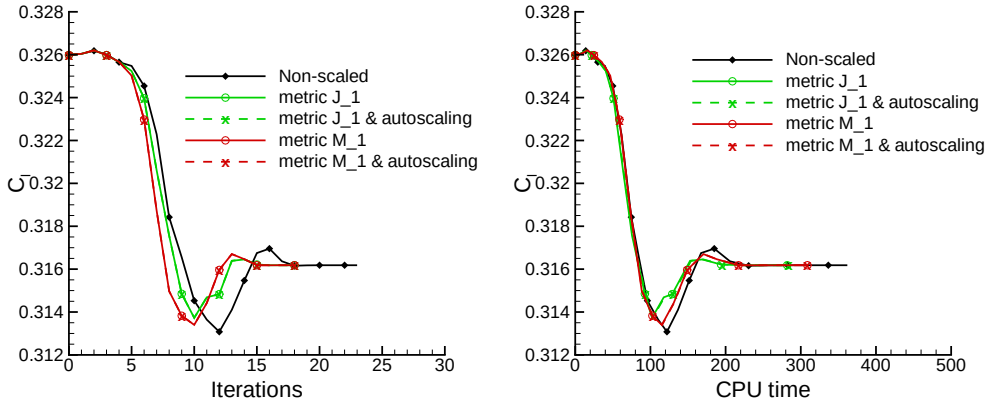
In Figures 5.5 and 5.6 we now illustrate the histories of the L_2-norm of

the non-scaled and scaled residuals obtained using inverse cell area, versus the number of iterations and CPU time, respectively. Convergence histories of other scaled computations are not presented because of very similar behaviours. The plots of both figures confirm the results presented so far. Rescaling techniques allow to improve the efficiency of DG computations by reducing the number of non-linear iterations and the computational effort needed to reach the convergence of each variable.

P_2



P_3



P_4

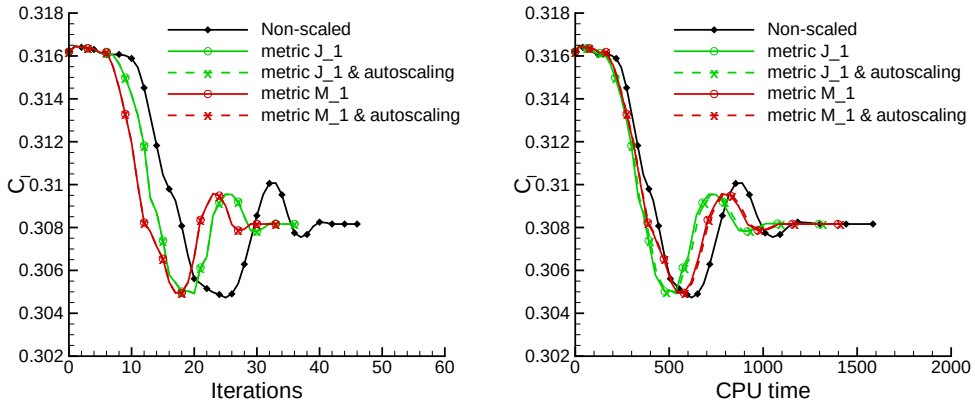
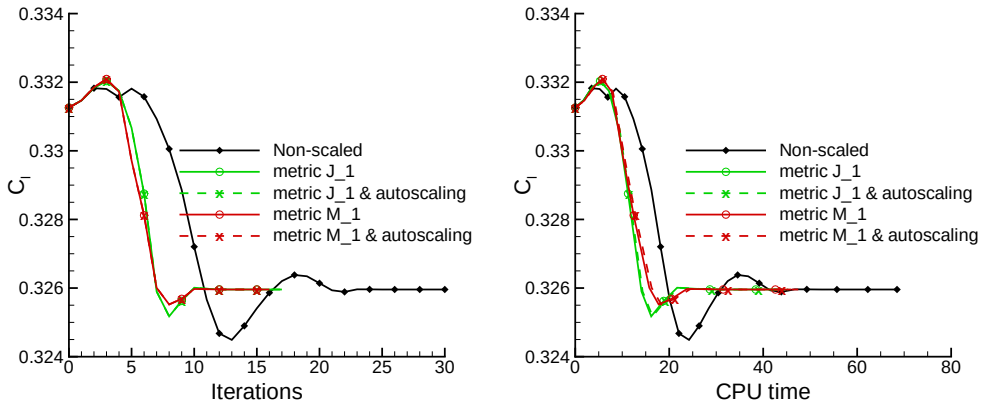
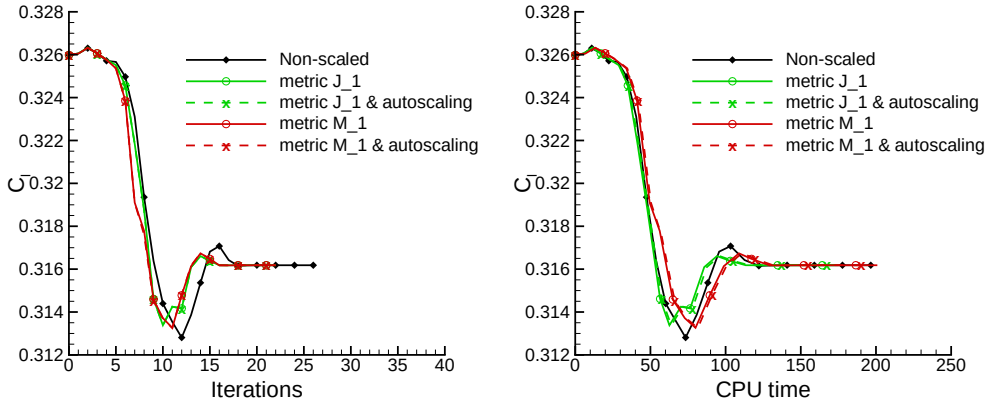


Figure 5.2: NACA0012 test case: C_l versus number of iterations (left column) and CPU time (right column) with and without scalings. Quadratic, P_2 (top row), cubic, P_3 (middle row) and quartic, P_4 (bottom row) elements. Linear residual reduction $= 10^{-6}$.

P_2



P_3



P_4

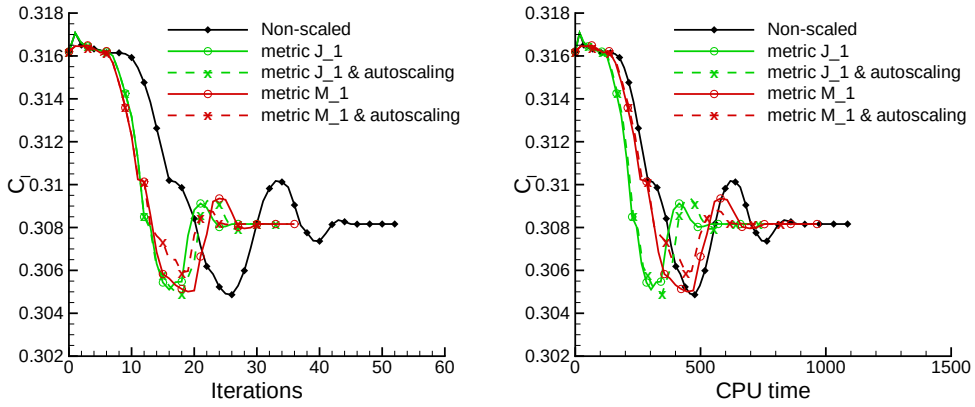
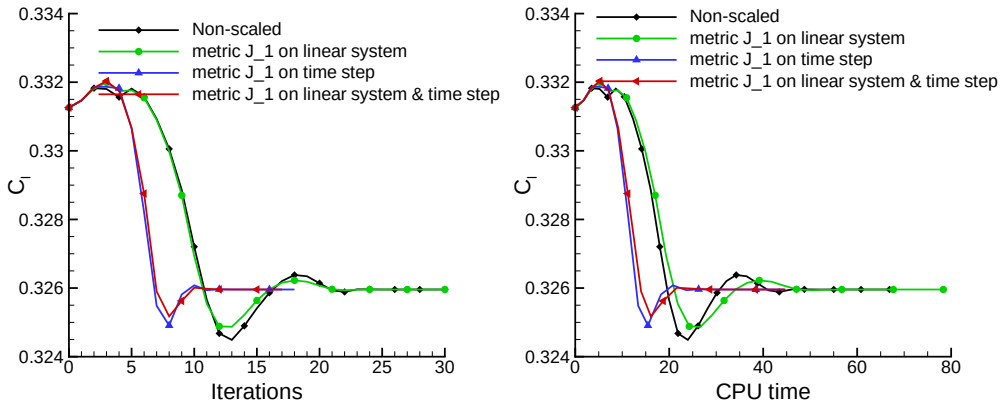
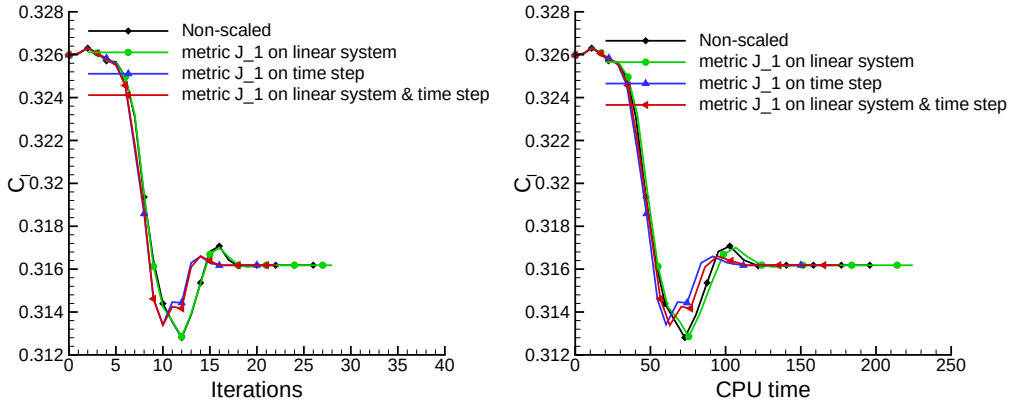


Figure 5.3: NACA0012 test case: C_l versus number of iterations (left column) and CPU time (right column) with and without scalings. Quadratic, P_2 (top row), cubic, P_3 (middle row) and quartic, P_4 (bottom row) elements. Linear residual reduction $= 10^{-1}$.

P_2



P_3



P_4

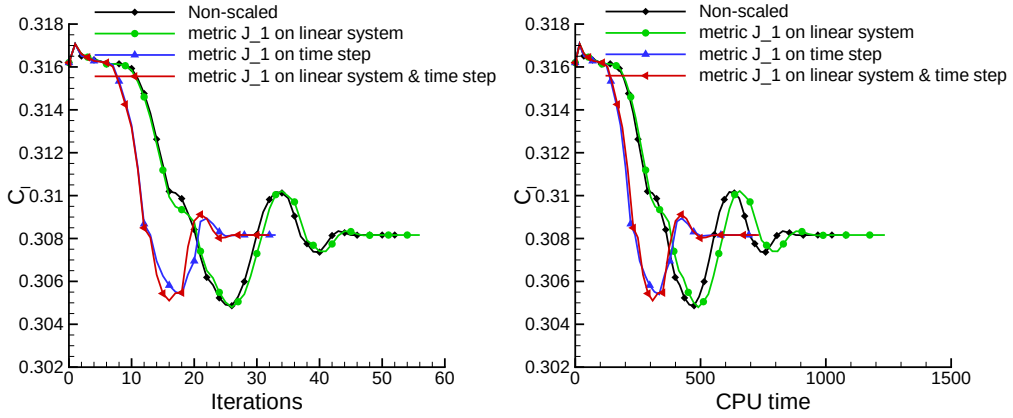


Figure 5.4: NACA0012 test case: C_l versus number of iterations (left column) and CPU time (right column) with and without metric scaling. Quadratic, P_2 (top row), cubic, P_3 (middle row) and quartic, P_4 (bottom row) elements. Linear residual reduction 10^{-1}.

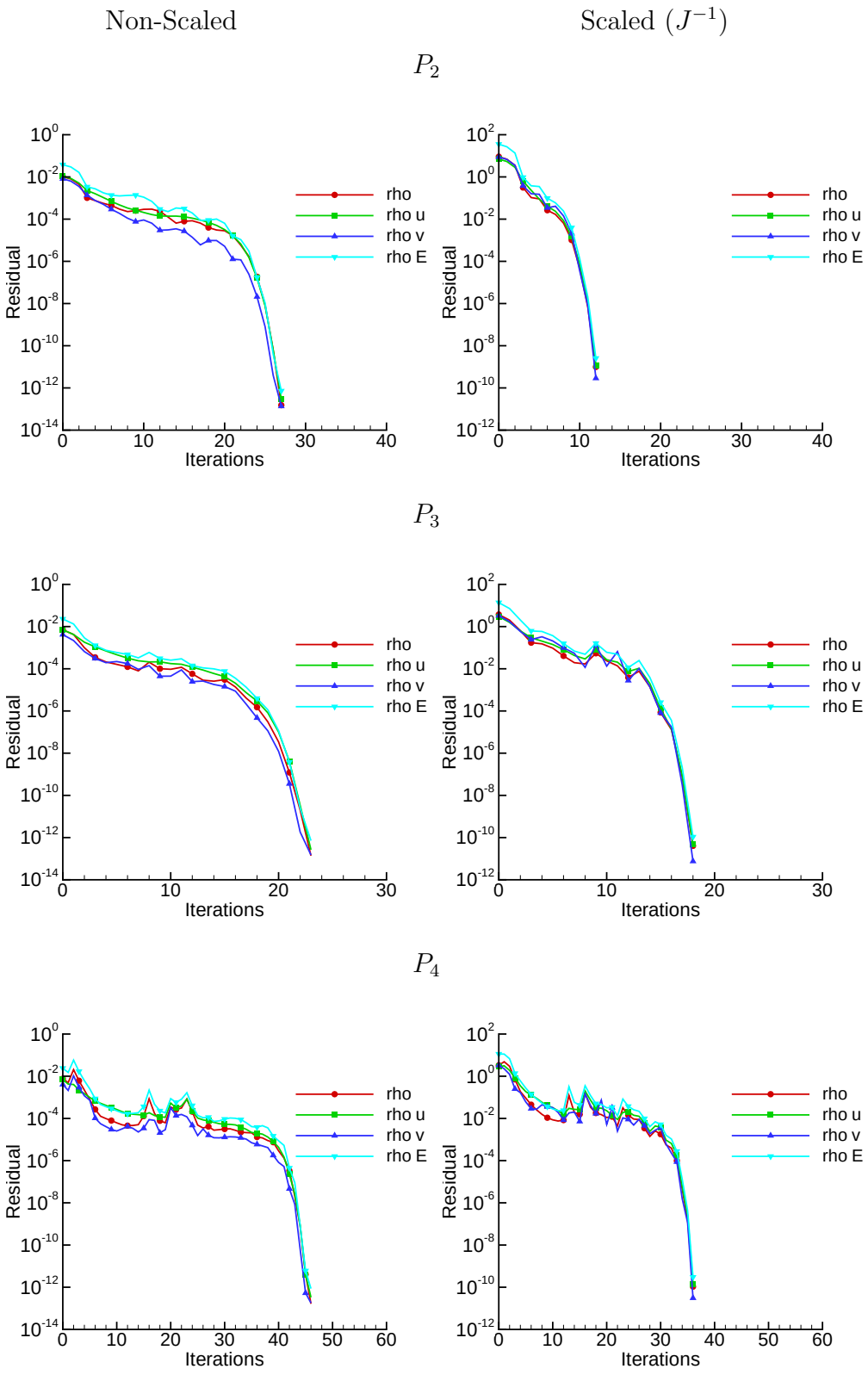


Figure 5.5: NACA0012 test case: history of L_2-norm of residuals versus number of iterations. Non-Scaled (left column) and Scaled (J^{-1}) (right column). Quadratic, P_2 (top row), cubic, P_3 (middle row) and quartic, P_4 (bottom row) elements. Linear residual reduction $= 10^{-6}$

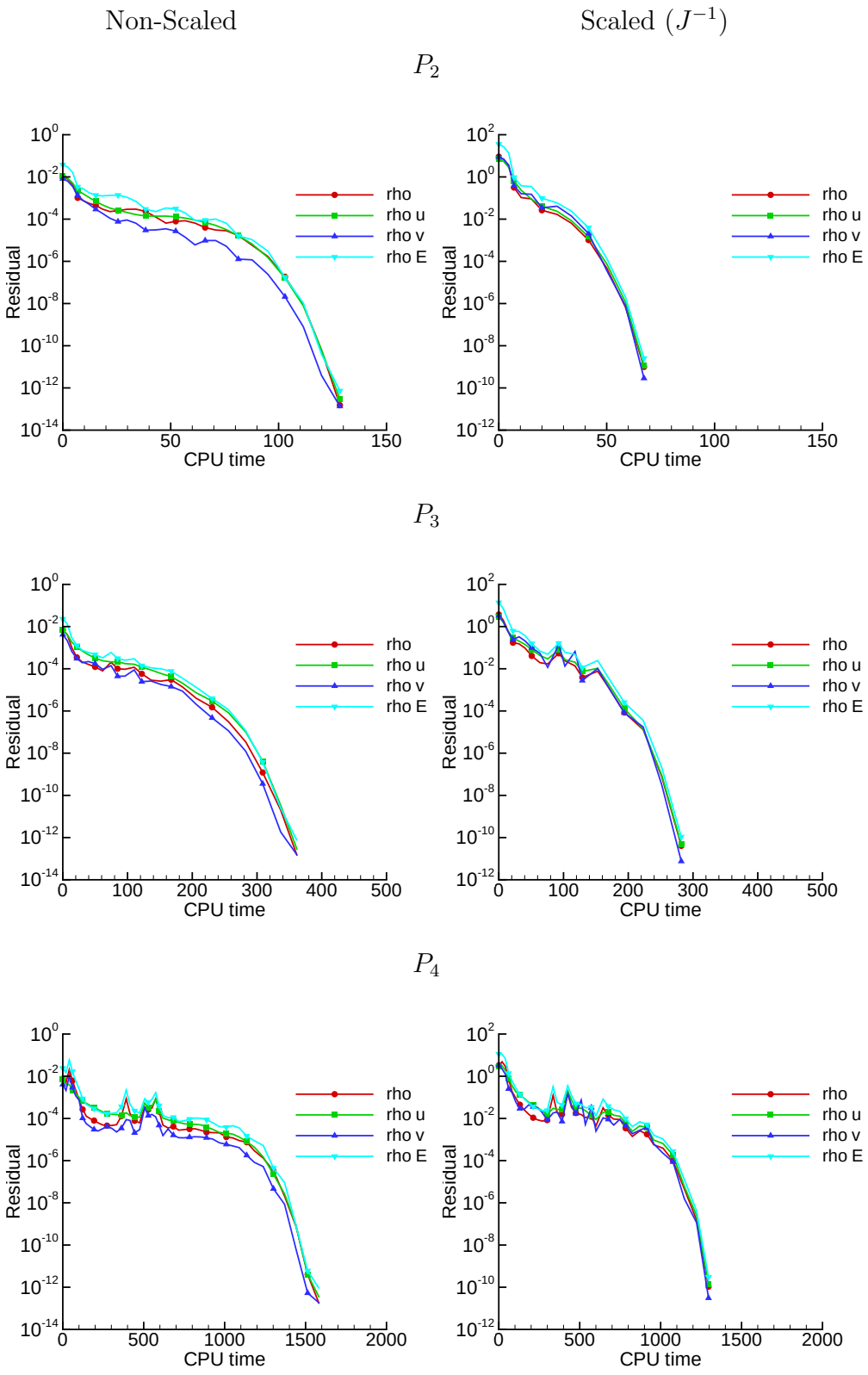


Figure 5.6: NACA0012 test case: history of L_2-norm of residuals versus CPU time. Non-Scaled (left column) and Scaled (J^{-1}) (middle column). Quadratic, P_2 (top row), cubic, P_3 (middle row) and quartic, P_4 (bottom row) elements. Linear residual reduction = 10^{-6}

5.3.2 RAE2822 airfoil, *Case 9*

In the second test case we consider a turbulent flow around a RAE2822 airfoil with the free-stream Mach number of $M_\infty = 0.73$, Reynolds number $Re = 6.5 \cdot 10^6$ and angle of attack $\alpha = 3.19°$, decreased to $\alpha = 2.79°$ in order to take into account the wind tunnel corrections. This is one of the standard turbulent test cases of the AGARD [111] and it was also considered in the ADIGMA project [27]. Our computations refer to the transonic conditions denoted by *Case 9*. A quadrilateral mesh of 8096 elements is used with lines represented by polynomials of degree 4, in order to represent the curved geometry, see Figure 5.7. A zero heat flux no slip boundary condition, where the ω_w is computed according to the projection discussed in Section 3.3.1, whereas at far-field the free-stream conditions with vortex correction are applied. At the inflow, k and ω are computed based on prescribed values of turbulence intensity, I_t, and turbulent viscosity ratio, $\frac{\mu_t}{\mu}$, set equal to 10^{-3} and 10^{-2}, respectively. For this test case, the linear systems at each Newton iteration are solved with 400 linear steps and one restart performed after 200 iterations. The linear solver is stopped once the linear residual is reduced by a factor of 10^{-8} or the maximum number of linear iterations is reached. The solutions are computed through a sequence of P_1, P_2 and P_3 approximations that starts from an initialization of free-stream values for P_0 elements.

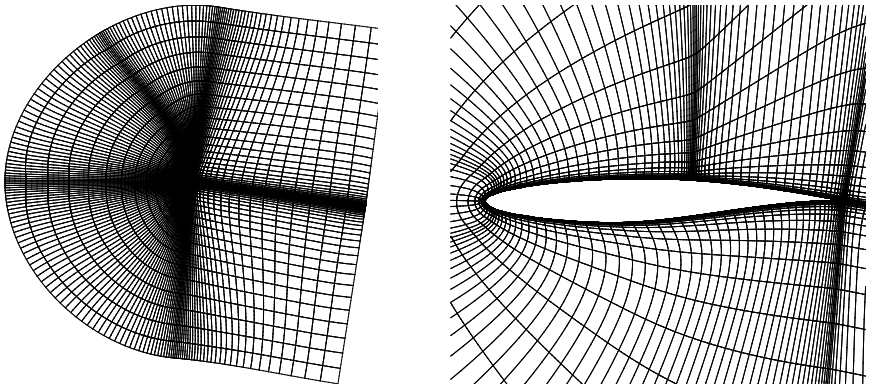


Figure 5.7: Computational grid, RAE2822 test case.

Figure 5.8 shows convergence histories of the lift coefficient obtained with non-scaled and scaled algorithms. A comparison between inverse cell area metric scaling, turbulent scaling, auto-scaling and their combination is presented. Note that, the inverse mass matrix scaling results are not shown since they do not lead to any improvement in the efficiency in comparison to the inverse cell area scaling. The computations are performed using the same initial CFL numbers, $CFL_0 = \{10, 8, 5\}$, for P_1, P_2 and P_3 elements, respectively. Overall, graphs show that, for each polynomial degree, lift coefficients converge to the same values independent of the scalings used. Furthermore, in the most of the cases, the use of scalings improves the efficiency of the scheme. In particular, for P_1 elements, scaled and non-scaled solutions exhibit similar convergence histories, except that for the inverse cell area scaling, which is less performant than the others due to instabilities caused by a too-big CFL_0. For P_2 elements we observe that the convergence history computed using the scaling of the turbulence model equations only is similar to that computed without scaling, whereas the use of the inverse cell area scaling allows a faster convergence to the reference value of the lift coefficient. Moreover, we note that, the coupling of the turbulent scaling and of the auto-scaling to the metric

scaling (full scaling) does not lead to any improvement in convergence rate and efficiency. Similarly, for P_3 solutions, turbulent scaling yields the same convergence behaviour of the non-scaled algorithm, whereas the other scalings allow to converge faster to the reference lift value. In particular, the C_l convergence is achieved with the lowest number of iterations and CPU time by using the metric scaling only. Finally, we remark that, using P_2 and P_3 elements, the scaled computations, except those performed using the turbulent scaling, allow to reduce the CPU time by a factor of 2 with respect to the non-scaled algorithm.

Figure 5.9 displays the history of residuals versus the number of implicit iteration steps of the backward Euler scheme with and without scalings, for linear, quadratic and cubic elements. The plots refer to computations performed using non-scaling and the more efficient scaling approaches. Overall, it is evident that scaling leads to a reduction in the number of non-linear iterations in comparison to the non-scaled algorithm, except for the P_1 metric scaled residuals that show spikes in the turbulent variable behaviours, which deteriorate the convergence process. The spikes in the residuals correspond to a large initial CFL number for the turbulent transport equations and are indicative of an instable solution. The addition of the turbulent scaling to the metric scaling ensures a stable P_1 solution, balancing the turbulent and the mean-flow residuals in the non-linear solver. Furthermore it is evident that auto-scaling does not lead to any improvement in the convergence of residuals. Finally, note that the difference between non-scaled and scaled Newton iterations becomes larger as the polynomial degree increases.

Figure 5.10 shows the histories of residuals versus CPU time (seconds). These plots point out that using P_2 and P_3 elements, scaling reduces the computational cost required to achieve convergence of the non-linear residuals as compared to the non-scaled algorithm, whereas P_1 computations are slightly more efficient without scaling. Furthermore, the saving of CPU time between non-scaled and scaled solutions increases when passing from linear to cubic

elements.

Figure 5.11 shows the effect of initial CFL number, CFL_0, on the convergence rate (left column) and efficiency (right column) of scaled and non-scaled computations using P_2 elements. The plots show that the metric scaling allows to converge in a lower number of non-linear iterations and exhibit a reduced computational time with respect to the other approaches up to $CFL_0 = 2$. Choosing higher initial CFL numbers, $CFL_0 = 5$, 10 and 15, results in a breakdown of the iteration process. The coupling of the metric scaling with turbulent and auto-scaling is less sensitive to numerical instabilities and allows to achieve the convergence for all the considered initial CLF numbers (except for $CFL_0 = 5$). The full scaled approach is shown to be more efficient than the non-scaled algorithm even if the gain in terms of number of iterations and CPU time decreases as the initial CFL number increases. Furthermore, it allowed to obtain the convergence for $CFL_0 = 15$, even if the non-scaled algorithm was more stable for $CFL_0 = 5$. These results, summarized in Tables 5.1 and 5.2, show that scaling influences both the evolution of the CFL number far from the solution (small CFL), improving the convergence rate to steady state, and the robustness of the solver.

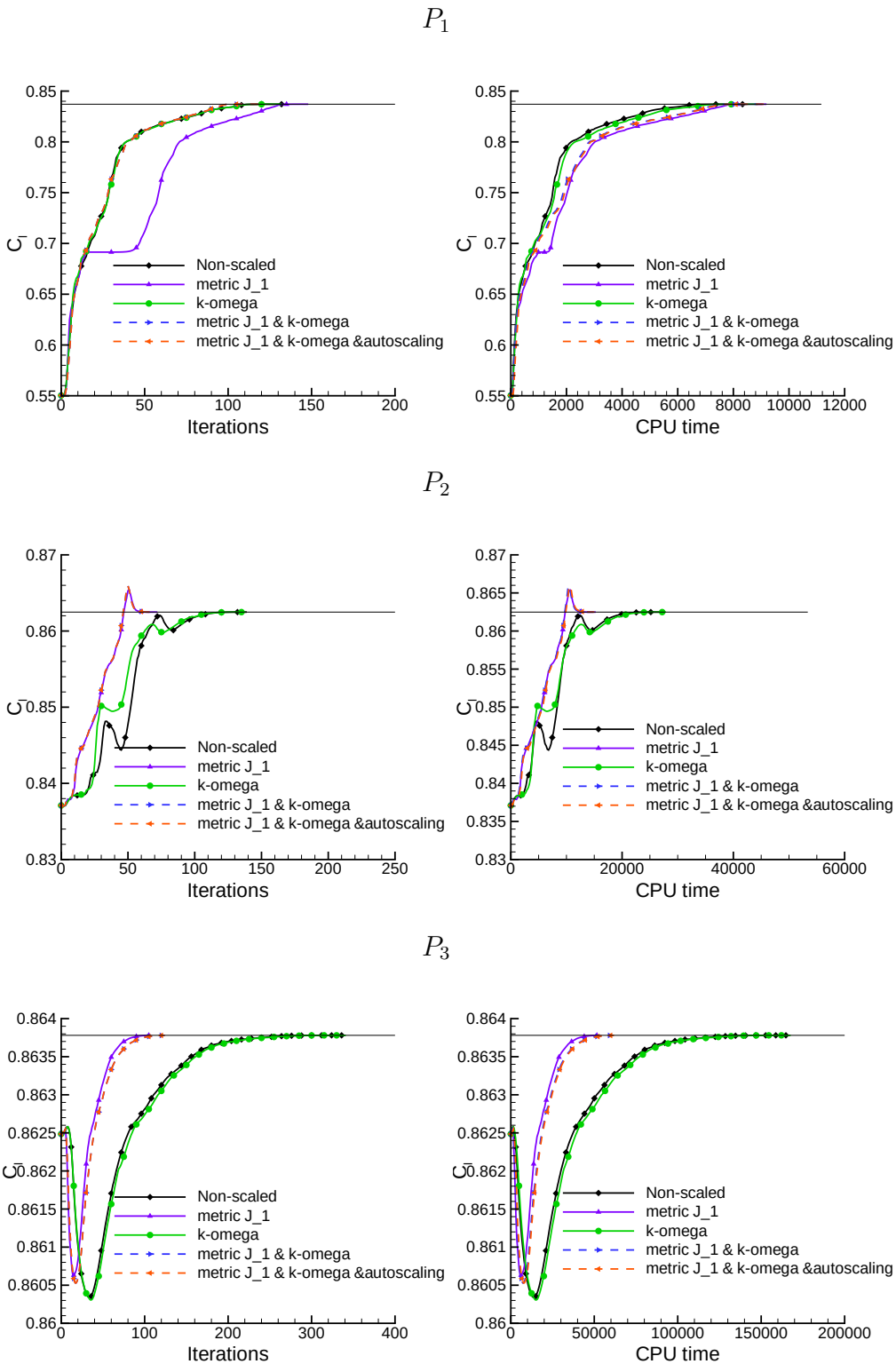


Figure 5.8: RAE2822 test case: C_l versus number of iterations (left column) and CPU time (right column) with and without scalings. Linear, P_1 (top row), quadratic, P_2 (middle row) and cubic, P_3 (bottom row) elements.

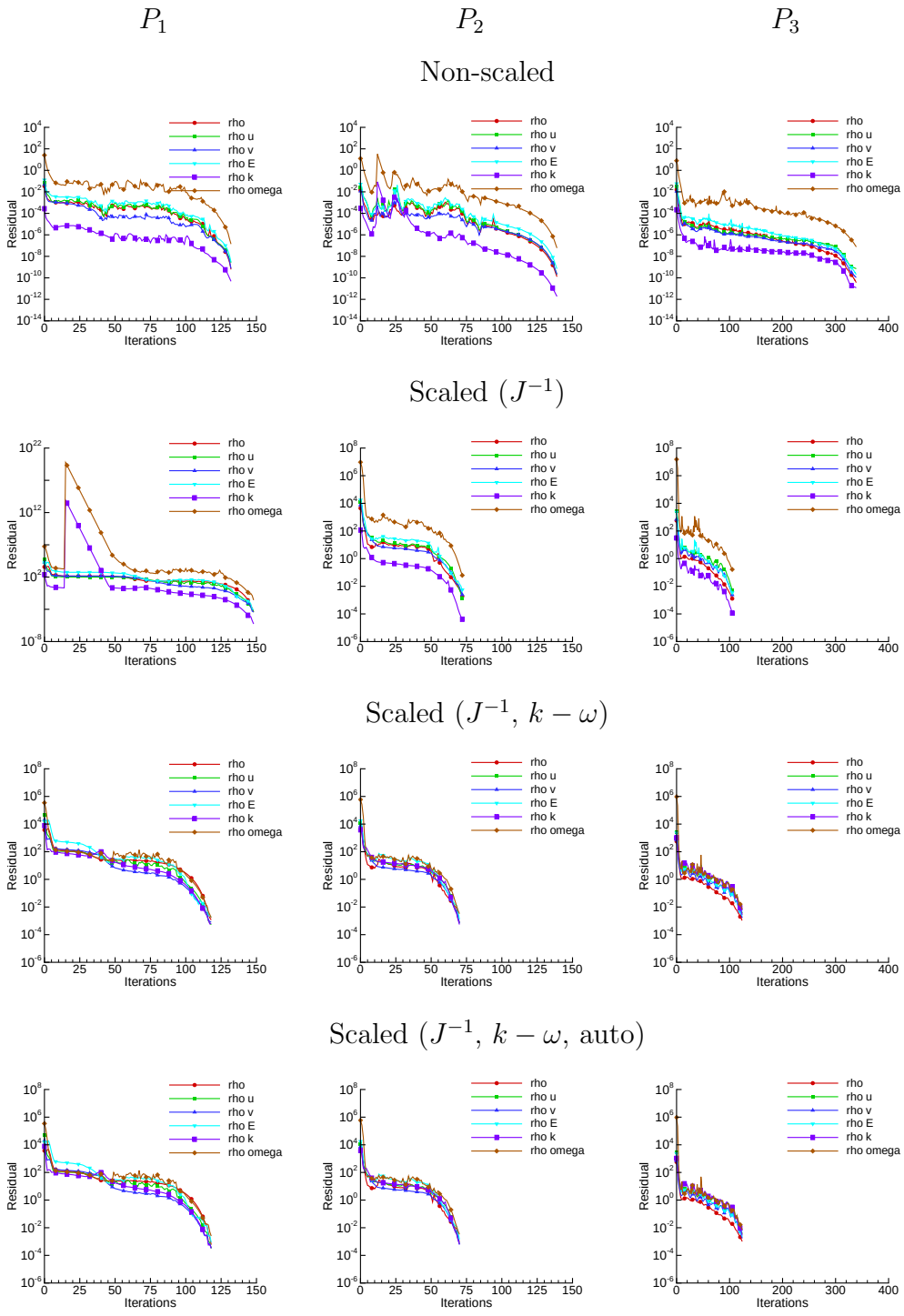


Figure 5.9: RAE2822 test case: history of L_2-norm of residuals versus number of iterations. Non-Scaled (first row) and Scaled by (J^{-1}) (second row), Scaled by ($J^{-1}, k\omega$) (third row) and Scaled by ($J^{-1}, k\omega$, auto-scaling) (last row). Linear, P_1 (left column), quadratic, P_2 (middle column) and cubic, P_3 (right column) elements.

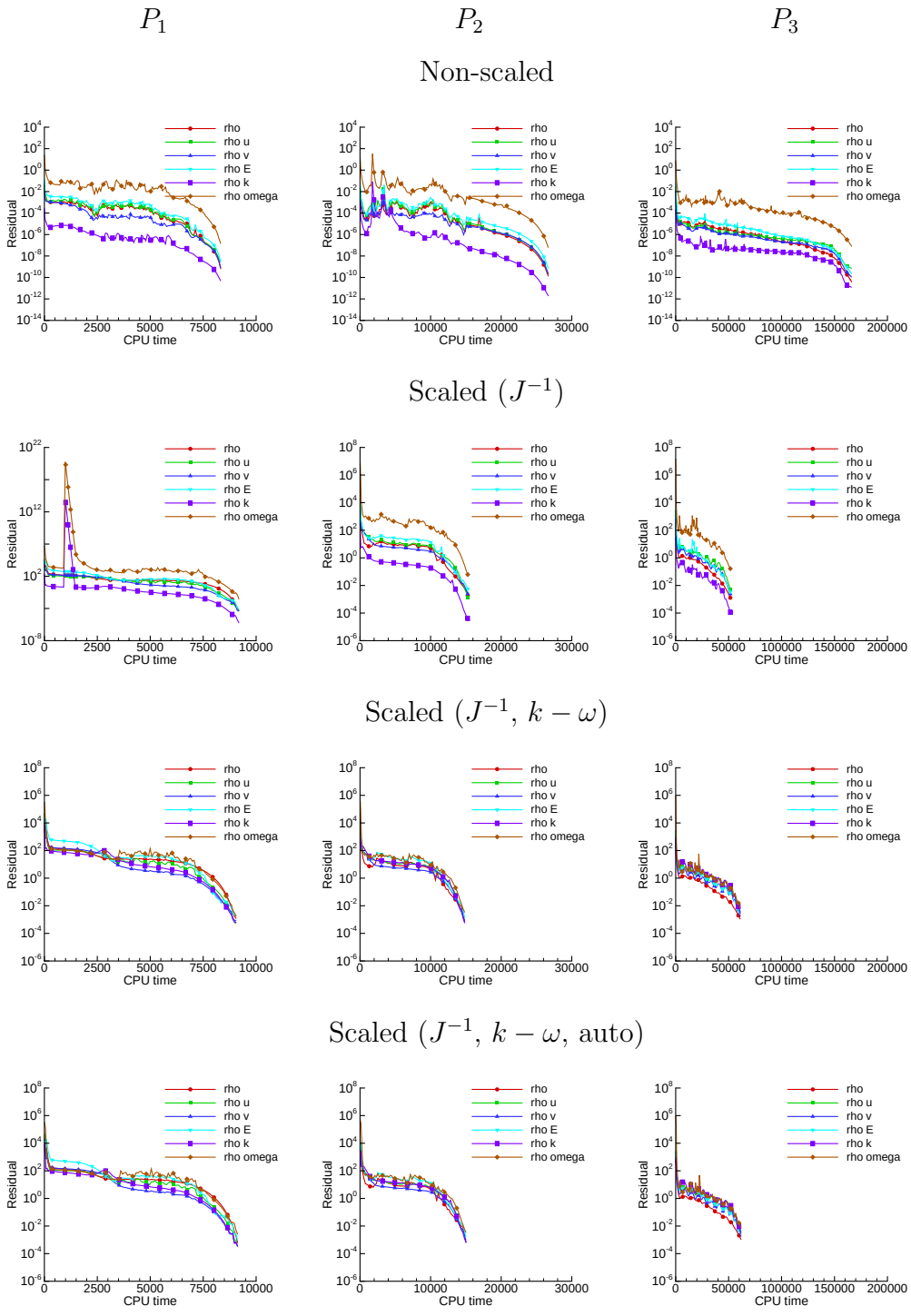


Figure 5.10: RAE2822 test case: history of L_2-norm of residuals versus CPU time. Non-Scaled (first row) and Scaled by (J^{-1}) (second row), Scaled by ($J^{-1}, k\omega$) (third row) and Scaled by ($J^{-1}, k\omega$, auto-scaling) (last row). Linear, P_1 (left column), quadratic, P_2 (middle column) and cubic, P_3 (right column) elements.

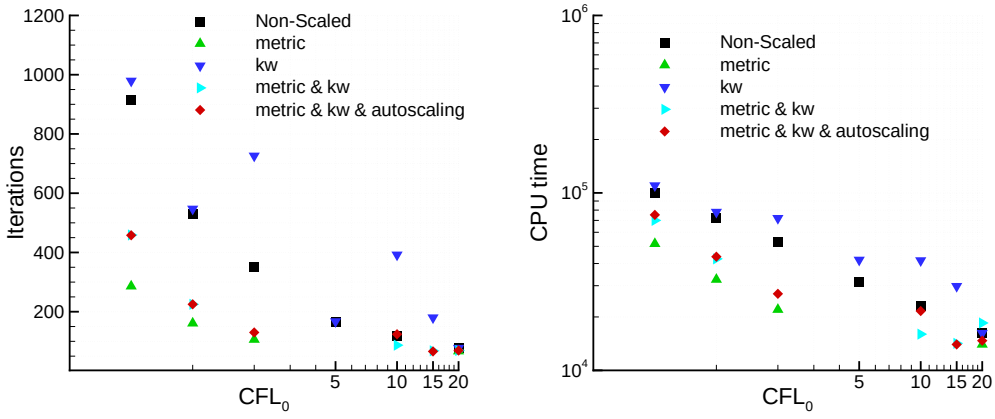


Figure 5.11: RAE2822 test case: number of iterations (left column) and CPU time (right column) required to achieve converged solution versus CFL_0 using non-scaled and scaled algorithms, P_2 elements.

	Iterations				
CFL_0	Non-Scal.	$\mathbf{J}^{-1}$	$k-\omega$	$\mathbf{J}^{-1}$ & $k-\omega$	Full-Scal.
0.5	913	286	979	459	458
1	531	161	547	225	225
2	351	106	726	X	130
5	164	X	167	X	X
10	119	X	392	87	124
15	X	X	180	68	66
20	78	67	77	69	69

Table 5.1: RAE2822 test case: number of iterations required to achieve converged P_2 solutions using non-scaled and scaled algorithms for various initial CFL numbers.

	CPU time				
CFL_0	Non-Scal.	$\mathbf{J}^{-1}$	$k-\omega$	$\mathbf{J}^{-1}$ & $k-\omega$	Full-Scal.
0.5	100000	51700	110000	70000	72500
1	72500	32500	78000	42500	43700
2	52700	22000	72000	X	27000
5	31500	X	41900	X	X
10	23000	X	41600	16000	21600
15	X	X	29800	14200	14000
20	16300	14000	16200	18500	14700

Table 5.2: RAE2822 test case: CPU time (sec) required to achieve converged P_2 solutions using non-scaled and scaled algorithms for various initial CFL numbers.

Conclusions

The goal of this research was to give a contribution to the development of a high-order accurate Discontinuous Galerkin finite element method (DG) for compressible all-speed flows.

This work was motivated by the increasing demand of industry to advance CFD-aided design and analysis procedures.

The problems faced with the standard industrial codes are:

- Low accuracy (formally second-order) for turbulence, aeroacoustics, and many viscosity dominant flows, such as boundary layer flows, vortical flows, shock-boundary layer interactions, heat flux transfers, etc.
- Difficulties to treat complex turbulent flows with simultaneous presence of high and low-speed regions.
- Extreme stiffness introduced by the turbulent transport equations.
- Poor conditioning of the algebraic system arising from the discretization of the governing equations on grids with a large range of element sizes.

The flow solver is based on the discretization of the compressible fully-coupled Reynolds-Averaged Navier-Stokes and $k-\omega$ turbulence model equations in conservative variables by means of a DG method, to preserve the high-order spatial accuracy for simulations in complex geometries at different flow regimes using unstructured grids. The Roe's approximate Riemann solver with the Harten's entropy-fix was introduced to compute the numerical fluxes at element interfaces. The semi-discrete equations were integrated in time by

the backward Euler scheme, using the restarted generalized minimum residual (GMRES) method to approximately solve the linear systems. The incomplete LU preconditioning was employed to accelerate the convergence of the linear solver.

The DG solution was extended to the incompressible limit by implementing a low Mach number preconditioning technique. A new preconditioner based on a modified version of the Turkel's preconditioning matrix was presented to take into account the proposed formulation of the governing equations. The preconditioning method employed in this work alters both the time-derivative terms of the system of equations and the dissipative terms of the numerical flux function (full preconditioning approach) in order to improve the accuracy and efficiency of the scheme for steady state computations. The performance of the method was demonstrated by computing laminar and turbulent aerodynamic test cases at variours low Mach numbers.

The results showed that accurate solutions on relatively coarse meshes can be computed, even without preconditioning, provided that high-order elements are used. Nevertheless, the accuracy of the numerical solution is always improved by preconditioning as compared to the non-preconditioned case. Furthermore, preconditioning was proven to enhance the convergence rate and to reduce the computational effort, improving the efficiency of the GMRES solver. The impact of preconditioning on the robustness of the numerical scheme was also discussed showing that, for complex turbulent applications, a proper setting of the preconditioning parameters reduces oscillations in the convergence history of the non-linear residuals improving the stability of the convergence process.

The scaling of the equations was adopted to overcome the ill-conditioning problems arising from both the use of non-uniform grids (metric scaling) and the inherent difference between turbulence and mean-flow variables (turbulent scaling). Furthermore, another kind of scaling, named auto-scaling, was im-

plemented to keep the maximum difference between linear residuals to within one order of magnitude. Since transonic test cases were computed to assess the effectiveness of the rescaling algorithms in the framework of DG schemes, an artificial viscosity term was added to the DG discretized equations for the non-linear control of oscillations at discontinuities. Furthermore, the standard DG discretization, based on the non-preconditioned eigenvalues and eigenvectors, was used due to the fact that the preconditioner reduces to the identity matrix for high speed flow computations.

For most of the results obtained by applying the rescaling techniques to inviscid and turbulent problems, it was demonstrated that this technique is effective. In particular, it was shown that the metric scaling significantly improves the convergence rate and efficiency, whilst turbulent scaling and auto-scaling poorly influence the performance of the scheme. However, when coupled with the metric scaling, turbulent and auto- scalings improve the robustness of the algorithm. In conclusion, our findings suggest that the best approach to improve the efficiency and robustness of the scheme is to use the full scaling approach. Since scalings impact on both the time step evolution and the solution of the linear system, a further investigation has been carried out to gain more insight into how this approach influences the performance of the flow solver. The results clearly showed that rescaling mainly contributes to the computational efficiency through the evolution of the time step, whereas its influence through the linear system is almost negligible. Basically, for the given GMRES parameters, metric scaling improves the performance of the CFL evolution strategy far from the solution (small CFL) allowing the non-linear iterations to converge in less iterations and CPU time as compared to the non-scaled computations. Future work should further explore the parameters in the method that could improve the benefits of the scaling. In particular, the control parameters determining the sequence of CFL numbers and the convergence of the GMRES solver, respectively.

Nota: *La presente tesi è cofinanziata con il sostegno della Commissione Europea, Fondo Sociale Europeo e della Regione Calabria. L'autore è il solo responsabile di questa tesi e la Commissione Europea e la Regione Calabria declinano ogni responsabilità sull'uso che potrà essere fatto delle informazioni in essa contenute.*

Appendix A

Primitive variables

In the following, we give the formulae and matrices for the transformation between conservative variables $\mathbf{u} = (\rho, \rho u_1, \rho u_2, \rho E, \rho k, \rho\widetilde{\omega})^T$ and primitive variables $\mathbf{q} = (p, u_1, u_2, T, k, \widetilde{\omega})^T$. Here ρ denotes the density, $\mathbf{v}^2 = (u_1, u_2)^T$ the velocity vector, E the total energy, k the turbulent kinetic energy and $\widetilde{\omega} = ln\,(\omega)$, where ω is the turbulence dissipation rate. The pressure p can be computed using the equation of state of a perfect gas, $p = \rho RT$. The gas constant R is given by the difference between the specific heat capacities at constant pressure, c_p, and constant volume, c_v, $R = c_p - c_v$. Furthermore, the total energy E is given by the relation $E = c_v T + \frac{1}{2}\mathbf{v}^2 + k$, where T denotes the temperature.

Given $\mathbf{u}$ in conservative variables we compute $\mathbf{q}$ in primitive ones as follows

$$\mathbf{q} = \begin{pmatrix} p \\ u_1 \\ u_2 \\ T \\ k \\ \widetilde{\omega} \end{pmatrix} = \begin{pmatrix} (\gamma - 1)\,\rho e \\ \frac{\rho u_1}{\rho} \\ \frac{\rho u_2}{\rho} \\ (\gamma - 1)\,\frac{\rho e}{\rho} \\ \frac{\rho k}{\rho} \\ \frac{\rho\widetilde{\omega}}{\rho} \end{pmatrix},$$

where the static specific energy e for a perfect gas is calculated from the conservative variables as:

$$\rho e = \rho E - \frac{1}{2\rho}\left[(\rho u_1)^2 + (\rho u_2)^2\right] - \rho k.$$

Given $\mathbf{q}$ in primitive variables we compute $\mathbf{u}$ in conservative ones as follows

$$\mathbf{u} = \begin{pmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho E \\ \rho k \\ \rho \widetilde{\omega} \end{pmatrix} = \begin{pmatrix} \frac{p}{T} \\ \frac{p}{T} u_1 \\ \frac{p}{T} u_2 \\ \frac{p}{(\gamma-1)} + \frac{p}{2T}\left(u_1^2 + u_2^2\right) + \frac{p}{T} k \\ \frac{p}{T} k \\ \frac{p}{T} \widetilde{\omega} \end{pmatrix}.$$

Thereby, the transformation matrix from conservative to primitive variables $\mathbf{\Gamma}$ is given by

$$\mathbf{\Gamma} = \frac{\partial \mathbf{u}}{\partial \mathbf{q}} = \begin{pmatrix} \rho_p & 0 & 0 & \rho_T & 0 & 0 \\ \rho_p u_1 & \rho & 0 & \rho_T u_1 & 0 & 0 \\ \rho_p u_2 & 0 & \rho & \rho_T u_2 & 0 & 0 \\ \rho_p H - 1 & \rho u_1 & \rho u_2 & \rho_T H + \rho c_p & \rho & 0 \\ \rho_p k & 0 & 0 & \rho_T k & \rho & 0 \\ \rho_p \widetilde{\omega} & 0 & 0 & \rho_T \widetilde{\omega} & 0 & \rho \end{pmatrix},$$

with

$$\rho_p = \frac{1}{T},$$
$$\rho_T = -\frac{p}{T^2} = -\frac{\rho}{T},$$

and

$$H = E + \frac{p}{\rho}.$$

Finally, the transformation matrix from primitive to conservative variables $\mathbf{\Gamma}^{-1}$ is given by

$$\mathbf{\Gamma}^{-1} = \frac{\partial \mathbf{q}}{\partial \mathbf{u}} = \begin{pmatrix} \frac{\rho c_p + \rho_T\left(H - k - q^2\right)}{c_p \rho \rho_p + \rho_T} & \frac{u_1 \rho_T}{c_p \rho \rho_p + \rho_T} & \frac{u_1 \rho_T}{c_p \rho \rho_p + \rho_T} & -\frac{\rho_T}{c_p \rho \rho_p + \rho_T} & \frac{\rho_T}{c_p \rho \rho_p + \rho_T} & 0 \\ -\frac{u_1}{\rho} & \frac{1}{\rho} & 0 & 0 & 0 & 0 \\ -\frac{u_2}{\rho} & 0 & \frac{1}{\rho} & 0 & 0 & 0 \\ \frac{\rho_p\left(q^2 - H + k\right)}{c_p \rho \rho_p + \rho_T} & \frac{u_1 \rho_p}{c_p \rho \rho_p + \rho_T} & \frac{u_2 \rho_p}{c_p \rho \rho_p + \rho_T} & \frac{\rho_p}{c_p \rho \rho_p + \rho_T} & -\frac{\rho_p}{c_p \rho \rho_p + \rho_T} & 0 \\ -\frac{k}{\rho} & 0 & 0 & 0 & \frac{1}{\rho} & 0 \\ -\frac{\widetilde{\omega}}{\rho} & 0 & 0 & 0 & 0 & \frac{1}{\rho} \end{pmatrix}.$$

Appendix B

Preconditioned eigenvectors

The preconditioned dissipation matrix of the Roe scheme is given by

$$\hat{\mathbf{A}}\left(\mathbf{u}^{+},\mathbf{u}^{-},\mathbf{n}\right)\Delta\mathbf{u}=\frac{\partial\mathbf{u}}{\partial\mathbf{q}}\mathbf{P}_{q}\left|{\mathbf{P}_{q}}^{-1}\frac{\partial\mathbf{q}}{\partial\mathbf{u}}\left(\frac{\partial\mathbf{F}_{c}}{\partial\mathbf{q}}\cdot\mathbf{n}\right)\right|\Delta\mathbf{q}=\frac{\partial\mathbf{u}}{\partial\mathbf{q}}\mathbf{P}_{q}\Big(\ \hat{\mathbf{T}}_{q}|\hat{\mathbf{\Lambda}}|\hat{\mathbf{T}}_{q}^{-1}\Big)\Delta\mathbf{q},$$

where the eigenvalues of the convective flux Jacobian are

$$\hat{\mathbf{\Lambda}}=\mathrm{diag}\left\{\mathbf{v}\cdot\mathbf{n},\mathbf{v}\cdot\mathbf{n},\mathbf{v}\cdot\mathbf{n},\mathbf{v}\cdot\mathbf{n},v'-a_{t}',v'+a_{t}'\right\},$$

and their associated eigenvectors, columns of the matrix $\hat{\mathbf{T}}_{q}$, are given by

$$\{0,0,0,0,0,1\}^{T},$$

$$\{-\frac{2T\rho}{2k+3T},\frac{6Tu_{1}\left(\beta^{2}-1\right)}{\left(2k+3T\right)\beta^{2}\left(2k\left(\gamma-1\right)+3T\gamma\right)},\frac{6Tu_{2}\left(\beta^{2}-1\right)}{\left(2k+3T\right)\beta^{2}\left(2k\left(\gamma-1\right)+3T\gamma\right)},0,1,0\}^{T},$$

$$\{\frac{2k\rho}{2k+3T},-\frac{6ku_{1}\left(\beta^{2}-1\right)}{\left(2k+3T\right)\beta^{2}\left(2k\left(\gamma-1\right)+3T\gamma\right)},-\frac{6ku_{2}\left(\beta^{2}-1\right)}{\left(2k+3T\right)\beta^{2}\left(2k\left(\gamma-1\right)+3T\gamma\right)},1,0,0\}^{T},$$

$$\{0,-n_{2},n_{1},0,0,0\}^{T},$$

$$\{\frac{\left(2k\left(\gamma-1\right)+3T\gamma\right)\rho}{\left(\left(\gamma-1\right)\left(2k+3T\right)\right)},-\frac{\left(3n_{1}\left(a_{t}'-\mathbf{v}\cdot\mathbf{n}+v'\right)\right)}{\left(\left(\gamma-1\right)\beta^{2}\left(2k+3T\right)\right)},-\frac{\left(3n_{2}\left(a_{t}'-\mathbf{v}\cdot\mathbf{n}+v'\right)\right)}{\left(\left(\gamma-1\right)\beta^{2}\left(2k+3T\right)\right)},1,0,0\}^{T},$$

$$\{\frac{\left(2k\left(\gamma-1\right)+3T\gamma\right)\rho}{\left(\gamma-1\right)\left(2k+3T\right)},\frac{3n_{1}\left(a_{t}'+\mathbf{v}\cdot\mathbf{n}-v'\right)}{\left(\gamma-1\right)\beta^{2}\left(2k+3T\right)},\frac{3n_{2}\left(a_{t}'+\mathbf{v}\cdot\mathbf{n}-v'\right)}{\left(\gamma-1\right)\beta^{2}\left(2k+3T\right)},1,0,0\}^{T},$$

with

$$v'=\frac{1}{2}\left(\beta^{2}+1\right)\left(\mathbf{v}\cdot\mathbf{n}\right),$$

$$a_{t}'=\frac{1}{2}\sqrt{\left(\mathbf{v}\cdot\mathbf{n}\right)^{2}\left(\beta^{2}-1\right)^{2}+4\beta^{2}{a_{t}}^{2}}.$$

List of Figures

List of Tables

Bibliography

[1] B. Van Leer. Flux-vector splitting for the Euler equations. *Technical Report 81-11, ICASE*, 1981.

[2] B. Van Leer. Upwind-difference methods for Aerodynamic problems governed by the Euler equations. *Lectures in Applied Mathematics*, 22, 1985.

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

[4] P. Roe. Characteristic-based schemes for the Euler equations. *Annual Review of Fluid Mechanics*, 18:337–365, 1986.

[5] B. Van Leer, J. Thomas, P. Roe, and R. Newsome. A comparison of numerical flux formulas for the Euler and Navier-Stokes equations. *AIAA Paper*, 87-1104, 1987.

[6] Z. J. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H.T. Huynh, N. Kroll, G. May, P.O. Persson, B. van Leer, M. Visbal. High-Order CFD methods: current status and perspective. *Int. J. Numer. Meth. Fluids*, 72(8):811–845, 2013.

[7] C.W. Shu and S. Osher. Efficient implementation of essentially nonoscillatory shock-capturing schemes. *J. Comput. Phys.*, 77(2):439–471, 1988.

[8] C.W. Shu and S. Osher. Efficient implementation of essentially nonoscillatory shock-capturing schemes. II. *J. Comput. Phys.*, 83(1):32–78, 1989.

[9] C. Johnson and J. Pitkarata. An Analysis of the Discontinuous Galerkin Method for a Scalar Hyperbolic Equation. *Mathematics of Computation*, 46:1–26, 1986.

[10] B. Cockburn, S.Y. Lin and C.W. Shu. TVB Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws III: One Dimensional Systems. *J. Comp. Phys.*, 84:90–113, 1989.

[11] B. Cockburn, G. Karniadakis, C.W. Shu. The development of discontinuous Galerkin methods. *Discontinuous Galerkin Methods*, Springer, 11:3–50, 1999.

[12] B. Cockburn, S. Hou and C.W. Shu. The Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws IV: The Multidimensional Case. *Mathematics of Computation*, 54(190):545–581, 1990.

[13] G. Jiang and C.W. Shu. On Cell Entropy Inequality for Discontinuous Galerkin Method for Hyperbolic Equations. *AIAA Journal*, 36:775–782, 1998.

[14] S.S Collis. The DG/VMS method for unified turbulence simulations. *AIAA Paper 2002-3124, 32nd AIAA Fluid Dynamics Conference 2002*

[15] T.J.R. Hughes, G. Scovazzi, P.B. Bochev, A. Buffa. A multiscale discontinuous Galerkin method with the computational structure of a continuous Galerkin method. *Comput. Methods Appl. Mwech. Engrg.*, 195:2761–2787, 2006.

[16] S. Gottlieb, A. Orszag. Numerical Analysis of Spectral Methods: Theory and Applications. *SIAM Philadelphia*, 1977.

[17] S. Gottlieb, JS. Hesthaven. Spectral Methods for Hyperbolic Problems *Journal of Computational and Applied Mathematics*, 128(1-2):83–131, 2001.

[18] A. Harten, B. Engquist, S. Osher, S. Chakravarthy. Uniformly high order essentially non-oscillatory schemes III. *Journal of Computational Physics*, 71:231, 1987.

[19] S. Lele. Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics*, 103:16–42, 1992.

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

[21] M. Visbal, D. Gaitonde. On the use of higher-order finite-difference schemes on curvilinear and deforming meshes. *Journal of Computational Physics*, 181(1):155–185, 2002.

[22] R. Abgrall, A. Larat, M. Ricchiuto. Construction of very high order residual distribution schemes for steady inviscid flow problems on hybrid unstructured meshes. *Journal of Computational Physics*, 230(11):4103–4136, 2011.

[23] T. Barth, P. Frederickson. High-order solution of the Euler equations on unstructured grids using quadratic reconstruction. *AIAA Paper*, 90–0013 1990.

[24] B. Cockburn, CW. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws ii: general framework. *Mathematics of Computation*, 52:411–435, 1989.

[25] F. Bassi, S. Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *Journal of Computational Physics*, 131:267–279, 1997.

[26] B. Van Leer, W.T. Lee, P.L. Roe. Characteristic time-stepping or local preconditioning of the Euler equations. *AIAA-1991-1552*, 260–282, 1991.

[27] N. Kroll, H. Bieler, H. Deconinck, V. Couaillier, H. van der Ven, K. Sorensen. ADIGMA - A European Initiative on the Development of Adaptive Higher-order Variational Methods for Aerospace Applications. *Notes on Numerical Fluid Mechanics and Multidisciplinary Design, Springer*, 113, 2010.

[28] N. Kroll. European project on Industrialization of High-Order Methods for Aeronautical Applications. *Proceedings of the ECCOMAS 2012, September 10-14, Vienna, Austria*, 2012.

[29] G. Volpe. Performance of compressible flow codes at low Mach numbers. *AIAA Journal*, 31:49–56, 1993.

[30] H. Guillard, C. Viozat. On the behavior of upwind schemes in the low Mach number limit. *Computers & Fluids*, 28:63–86, 1999.

[31] T.T. Chisholm, DW. Zingg. A Jacobian-free Newton-Krylov algorithm for compressible turbulent fluid flows. *Journal of Computational Physics*, 228:3490–3507, 2009.

[32] T.T. Chisholm. A fully coupled Newton-Krylov solver with a one-equation turbulence model. *PhD thesis, Insitute for Aerospace Studies, University of Toronto*, 2005.

[33] J. Blazek. Computational Fluid Dynamics: Principles and Applications. *Elsevier Science*, 460, Hardbound, 2001.

[34] W.A. Mulder, B. van Leer. Experiments with implicit upwind methods for the Euler equations. *Journal of Computational Physics*, 59(2):232–246, 1985.

[35] H.M. Bucker, B. Pollul, A. Rasch. On CFL Evolution Strategies for Implicit Upwind Methods in Linearized Euler Equations. *International Journal for Numerical Methods in Fluids*, 59(1):1–18, 2009.

[36] Y. Saad and M.H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Stat. Comput., 7:856–869, 1986. doi:10.1137/0907058

[37] I. Babuska, B.A. Szabo, and I.N. Katz. The p-Version of the Finite Element Method. *SIAM J. Numer. Anal.*, 18(3), 515–545, 1981.

[38] J. Ekaterinaris. High-order accurate, low numerical diffusion methods for aerodynamics. *Progress in Aerospace Sciences*, 41:192–300, 2005.

[39] P. Houston and E. Suli. A Note on the Design of hp-Adaptive Finite Element Methods for Elliptic Partial Differential Equations. *Computer Methods in Applied Mechanics and Engineering* 194(2-5):229-243, 2005.

[40] T. Leicht and R. Hartmann. Error Estimation and *hp*–Adaptive Mesh Refinement for Discontinuous Galerkin Methods. *Adaptive High-Order Methods in Computational Fluid Dynamics*, 2(3):67–94, World Science Books. Ed. Z. J. Wang, 2011.

[41] Z.J. Wang. High-order methods for the Euler and Navier-Stokes equations on unstructured grids. *Progress in Aerospace Sciences*, 43:1–41, 2007.

[42] W.H. Reed, T.R. Hill. Triangular mesh methods for the neutron transport equation. 1em Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.

[43] B. Cockburn, G. Karniadakis, and C. Shu. Discontinuous Galerkin methods: theory, computation and applications. *Lecture Notes in Computational Science and Engineering, Springer*, 11, 2000.

[44] F. Bassi, L. Botti, A. Colombo, A. Ghidoni, S. Rebay. Discontinuous Galerkin for Turbulent Flows. *Adaptive High-Order Methods in Computational Fluid Dynamics, Advances in Computational Fluid Dynamics*, World Science Books. Ed. Z. J. Wang, 2011.

[45] R. Hartmann. Higher-order and adaptive discontinuous Galerkin methods with shock-capturing applied to transonic turbulent delta wing flow. *Int. J. Numer. Meth. Fluids,* 72:883–894, 2013.

[46] R. Hartmann and P. Houston. Symmetric interior penalty DG methods for the compressible NavierStokes equations I: method formulation. *International Journal of Numerical Analysis and Modeling*, 3(1):1–20, 2006.

[47] B. Engquist and A. Majda. Absorbing boundary conditions for numerical simulation of waves. *Mathematics of Computations*, 31:629–651, 1997.

[48] J. Weiss and W.A. Smith. Preconditioning applied to variable and constant density flows. *AIAA Journal*, 33:2050–2057, 1995.

[49] Y.H. Choi and C.L. Merkle. The Application of Preconditioning in Viscous Flows. *Journal of Computational Physics*, 105:207–223, 1993.

[50] D. Givoli. Non-reflecting boundary conditions. *Journal of Computational Physics*, 94:1–29, 1991.

[51] A. Bayliss and E. Turkel. Far field boundary conditions for compressible flow. *Journal of Computational Physics*, 48:182–199, 1982.

[52] E. Turkel. Preconditioning techniques in computational fluid dynamics. *Annual Review of Fluid Mechanics*, 31:385–416, 1999.

[53] X.S. Li and C.W. Gu. An All-Speed Roe-type scheme and its asymptotic analysis of low Mach number behaviour. *Journal of Computational Physics*, 227(10):5144–5159, 2008.

[54] A.J. Chorin. A numerical method for solving incompressible viscous flow problems. *Journal of Computational Physics*, 135:118–125, 1997.

[55] F. Rieper. A low-Mach number fix for Roe's approximate Riemann solver. *Journal of Computational Physics*, 230(13):5263–5287, 2011.

[56] C. Rossow. A blended pressure-density based method for the computation of incompressible and compressible flows. *Journal of Computational Physics*, 185(2):375–398, 2003.

[57] E. Turkel. Preconditioned Methods for solving the Incompressible and Low Speed Compressible Equations. *Journal of Computational Physics*, 72:277–298, 1987.

[58] W.T. Lee. Local Preconditioning of the Euler equations. *PhD thesis, University of Michigan*, 1991.

[59] R. Hartmann, J. Held, T. Leicht and F. Prill. Discontinuous Galerkin Methods for Computational Aerodynamics – 3D Adaptive Flow Simulation with the DLR PADGE Code., *Aerosp. Sci. Technol.*, 14:512–519, 2010.

[60] R. Hartmann. Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier-Stokes equations. *Int. J. Numer. Meth. Fluids,* 51(9-10):1131–1156, 2006.

[61] W. Bangerth, R. Hartmann, and G. Kanschat. deal.II – A General Purpose Object Oriented Finite Element Library. *ACM Transactions on Mathematical Software*, 33(4) 24:1–24:27, 2007.

[62] F. Bassi, A. Crivellini, D.A. Di Pietro, and S. Rebay. A high-order Discontinuous Galerkin solver for 3D aerodynamic turbulent flows. *Proceedings of the Conference ECCOMAS CFD*, 2006

[63] B. Gustafsson. Far field boundary conditions for time-dependent hyperbolic systems. *SIAM Journal on Scientific and Statistical Computing*, 9(5):812–828, 1988.

[64] F. Bassi and S. Rebay. High order accurate discontinuous finite element solution of the 2d Euler equations. *Journal of Computational Physics*, 138:251–285, 1997.

[65] F. Bassi and S. Rebay. High-order accurate discontinuous Galerkin methods in computational fluid dynamics: from model problems to complex turbulent flows - Part 1. *In VKI LS 2008-08, 35th CFD/Adigma Course on Very High Order Discretization Methods: October 13 - 17, Deconinck H (ed.). von Karman institute: Rhode Saint Genese*, 1-31, 2008.

[66] F. Bassi and A. Nigro. A discontinuous Galerkin method for inviscid low Mach number flows. *Journal of Computational Physics* 228(11):3996–4011, 2009.

[67] P.R. Spalart and S.R. Allmaras. A One-Equation Turbulence Model for Aerodynamic Flows. *Recherche Aerospatiale, No. 1,* 5–21, 1994.

[68] T. Gatski and J. Bonnet. Compressibility, Turbulence and High Speed Flow. *Elsevier, Amsterdam*, 2009.

[69] K. Hanjalic. Will RANS Survive LES? A View of Perspectives. *Journal of Fluids Engineering*, 127:831–839, 2005.

[70] C.H. Son, T.A. Shethaji, C.J. Rutland, H. Barths, A. Lippert, S.H. El Tahry. Application of nonlinear turbulence models in an engine-type flow configuration. *International Journal of Engine Research*, 8:449–464, 2007.

[71] S.H. El Tahry, D.C. Haworth, Directions in turbulence modeling for in-cylinder Flows in reciprocating IC engines. *AIAA Journal Prop. and Power*, 8:1040–1048, 1992.

[72] S.H. El Tahry, D.C. Haworth. A perspective on the state-of-the-art in IC engine combustion modeling. *Proc. SIAM Sixth International Conference on Combustion*, New Orleans - LA, 1996.

[73] M. Germano, U. Piomelli, P. Moin, and W.H. Cabot. A dynamic sub-grid scale eddy viscosity model. *Physics of Fluids*, A(3):1760–1765, 1991.

[74] P.R. Spalart. Comments on the feasibility of LES for wing and on a hybrid RANS/LES approach, *1st ASOSR CONFERENCE on DNS/LES. Arlington, TX*, 1997.

[75] F. Bassi, L. Botti, A. Colombo, C. De Bartolo. Implict high-order discontinuous Galerkin solution of turbulent flows with an explicit algebraic Reynolds stress model. *ECCOMAS 2012* J. Eberhardsteiner *et al.* (eds.) Vienna, Austria, September 10-14, 2012

[76] KY. Chien. Predictions of Channel and Boundary-Layer Flows with a Low-Reynolds-Number Turbulence Model. *AIAA Journal*, 20(1):33–38, 1982.

[77] DC. Wilcox. Turbulence Modeling for CFD. *3rd edition, DCW Industries, Inc., La Canada CA*, 2006.

[78] DC. Wilcox, Formulation of the k-omega Turbulence Model Revisited. *AIAA Journal*, 46(11):2823–2838, 2008.

[79] F. Menter. Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications. *AIAA Journal*, 32(8):1598–1605, 1994.

[80] F. Menter. Improved two-equation $k-\omega$ turbulence models for aerodynamic flows. *NASA Technical Memorandum 103975*, 1992.

[81] F. Bassi, A. Crivellini, S. Rebay, M. Savini. Discontinuous Galerkin solution of the Reynolds-averaged NavierStokes and $k-\omega$ turbulence model equations. *Computers & Fluids*, 34:507–540, 2005.

[82] S. Schoenawa and R. Hartmann. Discontinuous Galerkin discretization of the Reynolds-averaged Navier-Stokes equations with the shear-stress transport model. *J. Comput. Phys.*, In review, 2014.

[83] T. Leicht and R. Hartmann. Error estimation and anisotropic mesh refinement for 3d laminar aerodynamic flow simulations. *J. Comput. Phys.*, 229(19):7344-7360, 2010.

[84] F. Ilinca and D. Pelletier. Positivity preservation and adaptive solution for the $k-\omega$ model of turbulence. *AIAA Journal*, 36(1):44–50, 1996.

[85] B. Eisfeld Implementation of Reynolds Stress Models into the DLR-FLOWer Code. *Institutsbericht, DLR-IB 124-2004/31, Report of the Institute of Aerodynamics and Flow Technology, Braunschweig*, ISSN 1614-7790, 2004.

[86] B. Eisfeld and O. Brodersen. Advanced Turbulence Modelling and Stress Analysis for the DLR-F6 Configuration. *AIAA Paper* 4727, June 2005.

[87] P. Persson and J. Peraire. Sub-Cell Shock Capturing for Discontinuous Galerkin Methods. *American Institute of Aeronautics and Astronautics*, 10.2514/6.2006-112, 2006.

[88] A. Burbeau and A. Sagaut and C.H. Bruneau A problem-independent limiter for high-order Runge-Kutta discontinuous Galerkin methods. *J. Comput. Phys.*, 169(1):111–150, 2001.

[89] F. Bassi, C. De Bartolo, R. Hartmann, A. Nigro. A discontinuous Galerkin method for inviscid low Mach number flow. *Journal of Computational Physics*, 228(11):3996–4011, 2009.

[90] Y. Colin, H. Deniau, J.F. Boussuge. A robust low speed preconditioning formulation for viscous flow computations. *Journal of Computational Physics*, 47(1):1–15, 2011.

[91] I. Yoo, E. Kwak, S. Lee, B.S. Kim, S.H. Park. Computational study on aerodynamics of long-span bridges. *Journal of Mechanical Science and Technology,* 23:802–813, 2009.

[92] D.L. Darmofal, P. Moinier, M.B Giles. Eigenmode analysis of boundary conditions for the one-dimensional preconditioned Euler equations. *Journal of Computational Physics* 160:369–384, 2000.

[93] P. Moinier, M.B. Giles. Compressible Navier-Stokes equations for low Mach number applications. *European Conference on Computational Fluid Dynamics ECCOMAS CFD 2001*, 2001.

[94] K. Hejranfar, R. Kamali-Moghadam. Preconditioned characteristic boundary conditions for solution of the preconditioned Euler equations at low Mach number flows. *Journal of Computational Physics*, 231:4384–4402, 2012.

[95] F. Rieper, G. Bader. The influence of cell geometry on the accuracy of upwind schemes in the low Mach number regime. *Journal of Computational Physics* 228:2918–2933, 2009.

[96] F. Rieper. On the dissipation mechanism of upwind-schemes in the low Mach number regime: a comparison between Roe and HLL. *Journal of Computational Physics* 229:221–232, 2010.

[97] H. Guillard. On the behavior of upwind schemes in the low Mach number limit. IV: P0 approximation on triangular and tetrahedral cells. *Computers & Fluids* 38:1969–1972, 2009.

[98] K. Wieghardt and W. Tillman. On the turbulent friction layer for rising pressure. *Technical Memorandum 1314, NACA, 1951.* 1951.

[99] D.E. Coles and E.A. Hirst. Computation of turbulent boundary layers. *1968 AFOSR-IFP-Stanford Conference. Stanford University*, 1969.

[100] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. Curfman McInnes, B. F. Smith, and H. Zhang. `PETSc` *Web page.* http://www.mcs.anl.gov/petsc, 2001.

[101] P. L. Roe and J. Pike. Efficient construction and utilization of approximate Riemann solutions. *North Holland Publishing, The Netherlands*, 1984.

[102] E. Turkel, V.N. Vatsa, R. Radespiel. Preconditioning methods for low-speed flows. *NASA Contractor Report 201605, ICASE Report No. 96-57.*

[103] A. Nigro, S. Renda, C. De Bartolo, R. Hartmann, F. Bassi. A high-order accurate discontinuous Galerkin finite element method for laminar low Mach number flows. *Int. J. for Numerical Methods in Fluids*, 72(1):43–68, 2013.

[104] A. Harten, J.M. Hyman. Self adjusting grid methods for one-dimensional hyperbolic conservation laws. *Journal of Computational Physics*, 50(2):235–269, 1983.

[105] M. Wallraff, T. Leicht, M. Lange-Hegermann. Numerical flux functions for Reynolds-Averaged Navier-Stokes and $k-\omega$ turbulence model computations with a line-preconditioned p-multigrid discontinuous Galerkin solver. *Int. J. Numer. Meth. Fluids*, 71:1055-1072, 2013.

[106] S. R. Allmaras, V. Venkatakrishnan, and F. T. Johnson. Farfield boundary conditions for 2-D airfoils. *43th AIAA Aerospace Sciences Meeting*, AIAA 2005-4711, 2005.

[107] F. Bassi, L. Botti, A. Colombo, A. Crivellini, N. Franchina, A. Ghidoni, S. Rebay. Very high-order accurate discontinuous Galerkin computation of transonic turbulent flows on aeronautical configurations, in N. Kroll, H. Bieler, H. Deconinck, V. Couallier, H. van der Ven, K. Sorensen (Eds.), ADIGMA - A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications. *Notes on Numerical Fluid Mechanics and Multidisciplinary Design, Springer*, 113:25–38, 2010.

[108] P. Birken, A. Meister. Stability of preconditioned finite volume schemes at low Mach numbers. *BIT*, 45:463-480, 2005.

[109] D. Unrau, DW. Zingg. Viscous airfoil computations using local preconditioning. *AIAA Paper*, 96:2088, 1996.

[110] R. Hartmann. Adaptive Finite Element Methods for the Compressible Euler Equations. *PhD thesis, University of Heidelberg*, 2002.

[111] P.H. Cook, M.A. Mc Donald, M.C.P. Firmin. Aerofoil RAE 2822, pressure distributions, and boundary layer and wake measurements. *Experimental Data Base for Computer Program Assessment, AGARD Report AR 138*, 1979.