An analysis of the performance of a high–order stabilised finite element method for simulating compressible flows

Ruben Sevilla, Oubay Hassan, Kenneth Morgan

Civil and Computational Engineering Centre (C^2EC), College of Engineering, Swansea University, Singleton Park, Swansea, SA2 8PP, Wales, UK.

Abstract

Stabilised finite element methods are now well established and are used in both industrial and commercial flow solvers. However, these methods have traditionally been restricted to linear approximations in space and they are rarely implemented within a high–order context. In the present work, the numerical performance of a higher–order approach is investigated for the simulation of some inviscid and laminar viscous two dimensional subsonic and transonic flows. The results produced clearly demonstrate the benefits that the higher–order approach has to offer, in terms of a reduction in both the degrees of freedom and, more importantly, the CPU time required, when compared with a low–order finite element or finite volume method.

Keywords: higher–order, stabilised finite elements, CPU time, compressible flow, streamline upwind/Petrov–Galerkin (SUPG)

1. Introduction

Second order finite volume methods remain the predominant numerical technique employed in industrial aerodynamic flow computations. However, the necessity of simulating high Reynolds number flows over complex configurations presents a major challenge for such methods, as a large number of degrees of freedom is generally required to obtain sufficiently accurate results.

Email addresses: R.Sevilla@swansea.ac.uk (Ruben Sevilla), O.Hassan@swansea.ac.uk (Oubay Hassan), K.Morgan@swansea.ac.uk (Kenneth Morgan)

During the last decade, there has been a great interest in developing efficient high–order methods for computational fluid dynamics. An example is the EU supported ADIGMA project [1], which was aimed at developing innovative adaptive high–order methods for the next generation of industrial aerodynamic flow solvers. The results of this project indicated that high–order methods are able to reduce drastically the required number of degrees of freedom but, normally, at the price of increasing the computational cost. It is worth noting that a large part of the research undertaken in ADIGMA was focused on high–order discontinuous Galerkin methods [2] and it was concluded that further research development was required before this approach could be employed as the basis for an industrial code [3].

Stabilised finite element methods, such as the streamline upwind/Petrov–Galerkin (SUPG) method [4] or the Galerkin/least–squares (GLS) method [5], are well established and already employed in certain industrial and commercial codes [6, 7]. However, stabilised finite element methods have, traditionally, been employed with linear elements [8]. A comparison of low and higher order stabilised finite elements for the incompressible Navier–Stokes equations is presented in [9], where it is shown that cubic elements are between six and seven times more efficient than linear elements. The combination of high–order approximations and stabilised methods for the compressible Navier–Stokes equations is considered in [10]. In a variety of test cases, the authors show the advantages that higher order approximations bring, in terms efficiency, for obtaining a given accuracy. In fact, it is worth remarking that this work showed the potential advantages of a high–order stabilised finite element solver with respect to high–order DG solvers in the context of the ADIGMA project.

In the present work, the SUPG formulation is considered and it is employed with continuous linear, quadratic or cubic polynomial approximations in space and a discontinuous approximation in time. The performance of the resulting space–time stabilised finite element formulation is studied for two dimensional steady inviscid and laminar viscous flow simulations. Linear and higher–order approximations are compared, in terms of the convergence behaviour of aerodynamic quantities of practical interest, such as lift and drag coefficients. The comparisons concentrate upon the number of degrees of freedom employed and the CPU time required. The results produced are encouraging and show that higher–order finite element approximations are competitive, not only with respect to linear (second order accurate) finite element approximations, but also with respect to second order finite volume methods.

The remainder of the paper is organised as follows. Section 2 details the Euler and Navier–Stokes equations governing the flow of a compressible fluid. Sections 3 and 4 briefly summarise the space–time SUPG formulation and its numerical implementation. In Section 5, a number of two dimensional inviscid and low Reynolds viscous subsonic and transonic flows are considered in order to analyse the performance of the method. Finally, Section 6 presents the main conclusions of the work that has been presented.

2. Compressible flow equations

The Navier–Stokes equations, governing unsteady two dimensional flow of a compressible viscous fluid, can, in the absence of external body forces, be written in the dimensionless conservative form

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}_i}{\partial x_i} - \frac{\partial \mathbf{G}_i}{\partial x_i} = \mathbf{0} \qquad\qquad i = 1,2$$

where (x_1, x_2) are cartesian coordinates and the summation convention is employed. The vector of conservation variables, $\mathbf{U}$, and the inviscid and viscous flux vectors, $\mathbf{F}_i$ and $\mathbf{G}_i$ respectively, are given by

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho v_1 \\ \rho v_2 \\ \rho E \end{pmatrix} \qquad \mathbf{F}_i = \begin{pmatrix} \rho v_i \\ \rho v_1 v_i + P\delta_{1i} \\ \rho v_2 v_i + P\delta_{2i} \\ (\rho E + P) v_i \end{pmatrix} \qquad \mathbf{G}_i = \begin{pmatrix} 0 \\ \tau_{i1} \\ \tau_{i2} \\ v_k \tau_{ki} + q_i \end{pmatrix}$$

In these expressions, ρ is the density of the fluid, $\boldsymbol{v} = (v_1, v_2)$ is the velocity vector, E is the total energy, P is the pressure, τ_{ij} are the components of the stress tensor, (q_1, q_2) is the heat flux vector and δ_{ij} is the Kronecker delta.

For a perfect gas, the equation of state

$$P = (\gamma - 1)\rho\Big(E - \frac{1}{2} v_i^2\Big)$$

completes the system of equations, where $\gamma = 1.4$ for air.

Employing the Stokes hypothesis, the components of the stress tensor are given by

$$\tau_{ij} = \frac{\mu}{\mathrm{Re}}\Big(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\Big) - \frac{2\mu}{3}\frac{\partial v_k}{\partial x_k}\delta_{ij}$$

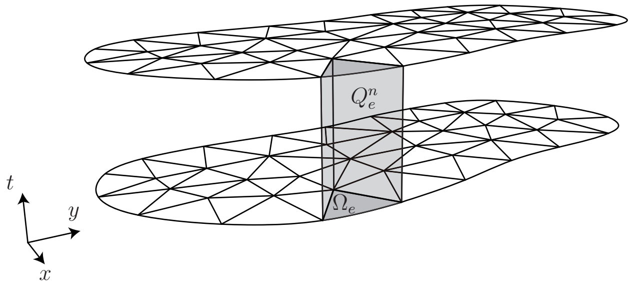


Figure 1: Schematic of a space–time slab.

where Re denotes the flow Reynolds number, and the fluid viscosity μ is assumed to vary with temperature according to Sutherland's law.

The Navier–Stokes equations can also be expressed in the non–conservative form

$$\frac{\partial \mathbf{U}}{\partial t} + \mathbf{A}_i \frac{\partial \mathbf{U}}{\partial x_i} - \frac{\partial}{\partial x_i}\left(\mathbf{K}_{ij}\frac{\partial \mathbf{U}}{\partial x_j}\right) = \mathbf{0} \tag{1}$$

where

$$\mathbf{A}_j = \frac{\partial \mathbf{F}_i}{\partial \mathbf{U}} \qquad\qquad \mathbf{K}_{ij}\frac{\partial \mathbf{U}}{\partial x_j} = \mathbf{G}_i$$

The non–conservative form of the Euler equations follows by setting $\mathbf{K}_{ij} = \mathbf{0}$.

3. Space–time stabilised finite element formulation

A recent review [8] describes the application of the space–time stabilised finite element method to the solution of the compressible Navier–Stokes equations. The formulation employed here is the streamline upwind/Petrov–Galerkin (SUPG) method [4] and it is combined with a discontinuous approximation in time [11].

The time domain $(0, T)$ is partitioned using N subintervals $I^n = (t^n, t^{n+1})$, where $n = 0, \ldots, N-1$. A space–time slab and its boundary are defined as $Q^n = \Omega \times I^n$ and $P^n = \partial\Omega \times I^n$ respectively, where Ω is the spatial domain. For each space–time slab Q^n, a regular partition of the spatial domain into non–overlapping elements is assumed, i.e. $\Omega = \bigcup_e \Omega_e$, such that $\Omega_i \bigcap \Omega_j = \emptyset$, for $i \neq j$, where a space–time element is defined as $Q_e^n = \Omega_e \times I^n$. A schematic representation of such a space–time element is depicted in Figure 1.

The solution at the interface between two space–time slabs is given by

$$\mathbf{V}(\boldsymbol{x}, t_{\pm}^n) = \lim_{\varepsilon \to 0^{\pm}} \mathbf{V}(\boldsymbol{x}, t^n + \varepsilon)$$

and the corresponding jump in the solution is represented as

$$[\![\mathbf{V}(t^n)]\!] = \mathbf{V}(t^n_+) - \mathbf{V}(t^n_-)$$

The solution and the weighting functions are approximated in terms of piecewise continuous polynomials of order p in space and are discontinuous in time, i.e. they belong to the space

$$\mathcal{V}_h = \bigcup_n \mathcal{V}_h^n, \; \mathcal{V}_h^n = \left\{ \mathbf{V}_h \mid \mathbf{V}_h \in \left(\mathcal{C}^0(Q^n)\right)^m, \mathbf{V}_h|_{Q_e^n} \in \left(\mathcal{P}_{\mathrm{p}}(Q_e^n)\right)^m, \forall \Omega_e \in \Omega \right\}$$

where $m = 4$ is the number of components of the solution vector $\mathbf{U}$. A space–time variational formulation for the compressible Navier–Stokes equations is: find $\mathbf{U}_h \in \mathcal{V}_h$ such that

$$\mathcal{A}_{\mathtt{Gal}}(\mathbf{U}_h,\mathbf{W}_h) - \mathcal{A}_{\mathtt{Jump}}(\mathbf{U}_h,\mathbf{W}_h) + \mathcal{A}_{\mathtt{SUPG}}(\mathbf{U}_h,\mathbf{W}_h) + \mathcal{A}_{\mathtt{Shock}}(\mathbf{U}_h,\mathbf{W}_h) = \mathcal{A}_{\mathtt{BC}}(\mathbf{U}_h,\mathbf{W}_h) \tag{2}$$

for all $\mathbf{W}_h \in \mathcal{V}_h$, where

$$\mathcal{A}_{\mathtt{Gal}}(\mathbf{U},\mathbf{W}) = \int_{Q^n} \left(\mathbf{W} \cdot \frac{\partial \mathbf{U}}{\partial t} - \frac{\partial \mathbf{W}}{\partial x_i} \cdot \mathbf{F}_i + \frac{\partial \mathbf{W}}{\partial x_i} \cdot \left(\mathbf{K}_{ij} \frac{\partial \mathbf{U}}{\partial x_j} \right) \right) dQ$$

$$\mathcal{A}_{\mathtt{Jump}}(\mathbf{U},\mathbf{W}) = \int_{\Omega} \mathbf{W} [\![\mathbf{U}(t^n)]\!] d\Omega$$

$$\mathcal{A}_{\mathtt{SUPG}}(\mathbf{U},\mathbf{W}) = \sum_e \int_{Q_e^n} \left(\left(\mathbf{A}_i \frac{\partial \mathbf{W}}{\partial x_i} \right) \cdot \boldsymbol{\tau} \left(\frac{\partial \mathbf{U}}{\partial t} + \mathbf{A}_i \frac{\partial \mathbf{U}}{\partial x_i} - \frac{\partial}{\partial x_i} \left(\mathbf{K}_{ij} \frac{\partial \mathbf{U}}{\partial x_j} \right) \right) \right) dQ$$

$$\mathcal{A}_{\mathtt{Shock}}(\mathbf{U},\mathbf{W}) = \sum_e \int_{Q_e^n} \boldsymbol{\nu}_{\mathtt{Shock}} \frac{\partial \mathbf{W}}{\partial x_i} \cdot \frac{\partial \mathbf{U}}{\partial x_i} dQ$$

and

$$\mathcal{A}_{\mathtt{BC}}(\mathbf{U},\mathbf{W}) = \int_{P^n} \mathbf{W} \cdot \Big(-\mathbf{F}_i + \mathbf{G}_i \Big) n_i dP$$

The terms $\mathcal{A}_{\mathtt{Gal}}$ and $\mathcal{A}_{\mathtt{BC}}$ are the terms that would appear in a standard Galerkin formulation. The time boundary integral $\mathcal{A}_{\mathtt{Jump}}$ is the jump condition that appears due to the use of an approximation that is discontinuous in time. This term imposes weakly the continuity of the solution across the boundary of the space–time slabs.

The term $\mathcal{A}_{\texttt{SUPG}}$ is the consistent SUPG stabilisation term, which is added to overcome the difficulties encountered when the standard Galerkin formulation is applied to convection dominated problems [12]. The second order spatial derivatives required for this term are computed here using a reconstruction procedure based upon an $\mathscr{L}^2$ projection [13]. The SUPG term incorporates a standard intrinsic time–scale matrix $\boldsymbol{\tau}$, which is a key ingredient in successful stabilised finite element formulations. Optimality of $\boldsymbol{\tau}$ has only been proved for the scalar convection–diffusion equation using linear finite elements [14]. For the complete Navier–Stokes system, a number of alternative definitions for the form of $\boldsymbol{\tau}$ have been proposed and compared. We have adopted the definition [15]

$$\boldsymbol{\tau} = \text{diag}(\tau^{\rho}, \tau^{v}, \tau^{v}, \tau^{e})$$

where

$$\tau^{\rho} = \left(\frac{1}{\tau_1^r} + \frac{1}{\tau_2^r}\right)^{-\frac{1}{r}} \qquad \tau^{v} = \left(\frac{1}{\tau_1^r} + \frac{1}{\tau_2^r} + \frac{1}{\tau_{3,v}^r}\right)^{-\frac{1}{r}} \qquad \tau^{e} = \left(\frac{1}{\tau_1^r} + \frac{1}{\tau_2^r} + \frac{1}{\tau_{3,e}^r}\right)^{-\frac{1}{r}}$$

Here, r is a free parameter, usually given the value 2, τ_1 represents the advective limit, τ_2 is the transient limit and $\tau_{3,v}$ and $\tau_{3,e}$ are the diffusive limits, with these limits defined as

$$\tau_1 = \left(\sum_{J=1}^{\texttt{n}_{\texttt{en}}} \left(\frac{c}{\|\nabla\rho\|}|\nabla\rho \cdot \nabla N_J| + |\boldsymbol{v} \cdot \nabla N_J|\right)\right)^{-1} \qquad \tau_2 = \frac{\Delta t}{2}$$

$$\tau_{3,v} = \frac{1}{\nu\big\|\nabla\|\boldsymbol{v}\|\big\|^2}\left(\sum_{J=1}^{\texttt{n}_{\texttt{en}}} \big|\nabla\|\boldsymbol{v}\|\cdot\nabla N_J\big|\right)^{-2} \qquad \tau_{3,e} = \frac{1}{\nu Pr\|\nabla T\|^2}\left(\sum_{J=1}^{\texttt{n}_{\texttt{en}}} |\nabla T\cdot\nabla N_J|\right)^{-2}$$

$\texttt{n}_{\texttt{en}}$ denotes the number of nodes per element, N_J denotes the shape function associated with node J of the mesh and $\Delta t = t^{n+1} - t^{n}$.

The term $\mathcal{A}_{\texttt{Shock}}$ is a Laplacian based artificial diffusion that provides a shock capturing capability. The artificial diffusion parameter, $\boldsymbol{\nu}_{\texttt{Shock}}$, which introduces the necessary diffusion to suppress oscillations in the vicinity of discontinuities is defined as [15]

$$\boldsymbol{\nu}_{\texttt{Shock}} = \nu_{\texttt{Shock}}\boldsymbol{I}$$

where

$$\nu_{\mathtt{Shock}} = \|\boldsymbol{Y}^{-1}\boldsymbol{Z}\| \left(\sum_{i=1}^{\mathtt{n_{sd}}} \left\| \boldsymbol{Y}^{-1} \frac{\partial \mathbf{U}}{\partial x_i} \right\|^2 \right)^{\beta/2-1} \left(\frac{1}{\|\nabla \rho\|} \sum_{J=1}^{\mathtt{n_{en}}} |\nabla \rho \cdot \nabla N_J| \right)^{-2}$$

where $\mathtt{n_{sd}} = 2$ is the number of spatial dimensions, $\boldsymbol{Y}$ is a diagonal matrix containing reference values for each of the vector solution components, β is a free parameter that enables control of the amount of added viscosity [15] and

$$\boldsymbol{Z} = \frac{\partial \mathbf{U}}{\partial t} + \mathbf{A}_i \frac{\partial \mathbf{U}}{\partial x_i}$$

This shock–capturing term is only consistent for inviscid simulations, as $\boldsymbol{Z}$ represents the residual of the Euler equation system. The definition

$$\boldsymbol{Z} = \frac{\partial \mathbf{U}}{\partial t} + \mathbf{A}_i \frac{\partial \mathbf{U}}{\partial x_i} - \frac{\partial}{\partial x_i} \left(\mathbf{K}_{ij} \frac{\partial \mathbf{U}}{\partial x_j} \right)$$

proposed in [16] is employed for the full Navier–Stokes equations.

4. Numerical solution

The solution is approximated using continuous piecewise polynomials of order p in space and a discontinuous approximation in time. On a reference space–time element $\widehat{Q}$, the approximate solution is expressed as

$$\mathbf{U}_h(\boldsymbol{\xi}, t') = \sum_{J=1}^{\mathtt{n_{en}}} N_J(\boldsymbol{\xi}) \boldsymbol{U}_J(t') \tag{3}$$

where N_J denotes the polynomial shape functions of order p, in the local coordinates $\boldsymbol{\xi}$, and the vector $\boldsymbol{U}$ denotes the time–dependent nodal values of the solution. By using a discontinuous approximation in time, with polynomials of order l, an implicit time integration algorithm of order $2l + 1$ is obtained [17]. For the steady state simulations that are reported in this paper, time is regarded as a relaxation parameter and piecewise constant approximations in time are used.

Substituting the approximation of equation (3) into the variational formulation of equation (2), and selecting the space of weighting functions to be equal to the space of approximation functions, results in the non–linear system of equations

$$\mathbf{R}(\boldsymbol{U}^{n+1}, \boldsymbol{U}^n) = \mathbf{0} \tag{4}$$

A Newton algorithm is applied, by linearising this non–linear system and truncating the Taylor series expansion of $\mathbf{R}$ at first order. The resulting equation is solved iteratively, with each Newton iteration requiring the solution of the sparse non–symmetric linear system

$$\frac{\partial \mathbf{R}(\boldsymbol{U}_k^{n+1}, \boldsymbol{U}^n)}{\partial \mathbf{U}} \Delta \boldsymbol{U}_k^{n+1} = -\mathbf{R}(\boldsymbol{U}_k^{n+1}, \boldsymbol{U}^n) \qquad (5)$$

Here, the k–th iterative approximation to $\boldsymbol{U}^{n+1}$ is denoted by $\boldsymbol{U}_k^{n+1}$, the initial condition is $\boldsymbol{U}_0^{n+1} = \boldsymbol{U}^n$ and

$$\Delta \boldsymbol{U}_k = \boldsymbol{U}_{k+1} - \boldsymbol{U}_k$$

The coefficient matrix and the right hand side vector in equation (5) are obtained by assembling the element contributions, given by

$$\begin{aligned}
\left(\frac{\partial \mathbf{R}^e}{\partial \mathbf{U}}\right)_{IJ} &= -\int_{\Omega_e} N_I \, \boldsymbol{I} N_J d\Omega + \Delta t \int_{\Omega_e} \left(-\frac{\partial N_I}{\partial x_i} \mathbf{A}_i N_J + \frac{\partial N_I}{\partial x_i} \mathbf{K}_{ij} \frac{\partial N_J}{\partial x_j} \right) d\Omega \\
&+ \Delta t \int_{\Omega_e} \left(\left(\mathbf{A}_i \frac{\partial N_I}{\partial x_i} \right) \boldsymbol{\tau} \left(\mathbf{A}_i \frac{\partial N_J}{\partial x_i} - \frac{\partial}{\partial x_i} \left(\mathbf{K}_{ij} \frac{\partial N_J}{\partial x_j} \right) \right) \right) d\Omega \\
&+ \Delta t \int_{\Omega_e} \boldsymbol{\nu}_{\texttt{Shock}} \frac{\partial N_I}{\partial x_i} \frac{\partial N_J}{\partial x_i} d\Omega + \Delta t \int_{\Gamma_e} N_I \left(\mathbf{A}_i N_J - \mathbf{K}_{ij} \frac{\partial N_J}{\partial x_j} \right) n_i d\Gamma
\end{aligned}$$

and

$$\begin{aligned}
\mathbf{R}_I^e &= -\int_{\Omega_e} N_I \, \llbracket (\mathbf{U}(t^n) \rrbracket d\Omega + \Delta t \int_{\Omega_e} \left(-\frac{\partial N_I}{\partial x_i} \mathbf{F}_i + \frac{\partial N_I}{\partial x_i} \left(\mathbf{K}_{ij} \frac{\partial \mathbf{U}}{\partial x_j} \right) \right) d\Omega \\
&+ \Delta t \int_{\Omega_e} \left(\left(\mathbf{A}_i \frac{\partial N_I}{\partial x_i} \right) \boldsymbol{\tau} \left(\mathbf{A}_i \frac{\partial \mathbf{U}}{\partial x_i} - \frac{\partial}{\partial x_i} \left(\mathbf{K}_{ij} \frac{\partial \mathbf{U}}{\partial x_j} \right) \right) \right) d\Omega \\
&+ \Delta t \int_{\Omega_e} \boldsymbol{\nu}_{\texttt{Shock}} \frac{\partial N_I}{\partial x_i} \frac{\partial \mathbf{U}}{\partial x_i} d\Omega + \Delta t \int_{\Gamma_e} N_I (\mathbf{F}_i - \mathbf{G}_i) n_i d\Gamma
\end{aligned}$$

In these equations, $\Gamma_e = \partial\Omega \bigcap \Omega_e$ and n_i is the i-th component of the outward unit normal vector to Γ_e. The integrals appearing here are evaluated using numerical quadrature in a local reference space–time element, with a mapping employed to relate the local and physical coordinates. The implementation considered here employs Gaussian quadratures that integrate exactly polynomials of order less or equal to $2\mathrm{p} + 1$, where p is the order

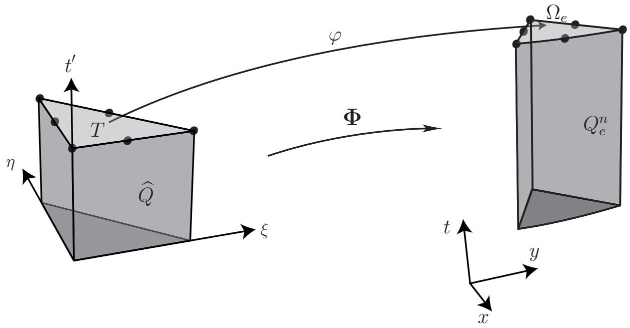


Figure 2: Schematic of the mapping between the reference space–time element ($\widehat{Q}$) and a physical space–time element (Q_e^n).

of the approximation. It is well known that this choice ensures the optimal convergence of isoparametric finite elements [18].

For steady state simulations, the spatial discretisation is the same for each time level and the space–time elements are prisms. In this case, the mapping between the local reference space–time element and a physical element can be defined using two independent mappings to handle the space and the time coordinates separately. The standard isoparametric mapping

$$\boldsymbol{\varphi}(\boldsymbol{\xi}) = \sum_{J=1}^{\mathtt{n_{en}}} \boldsymbol{x}_J N_J(\boldsymbol{\xi})$$

is used for the space coordinates. A schematic representation of the space–time mapping is represented in Figure 2, for the case of a quadratic approximation in space with a triangular element.

The linear system of equation (5) is solved approximately using a generalised minimum residual (GMRES) method [19], with an incomplete LU (ILU) factorisation used as a pre–conditioner to enhance the convergence [20].

5. Numerical examples

The numerical performance of the stabilised SUPG formulation, using both linear and higher–order approximations in space and a piecewise constant approximation in time, is now analysed for a number of steady–state compressible flow test cases. At each time step, as time accuracy is not important, the non–linear system of equations is approximated by using just one Newton iteration and the resulting linear system is solved, using GMRES, until the norm of the residual attains a relative tolerance of less than 10^{-2}. For each example, the solution is advanced in time using the maximum Courant number that produces stable results for each order of approximation, e.g. typical values for linear, quadratic and cubic approximations are 20, 15 and 10 respectively. The Courant number is defined as

$$C = 2\frac{\mathrm{p}^2 \Delta t}{h^2} \max\left(2\mu/\rho, \kappa/c_v\rho\right) + \frac{\mathrm{p}\Delta t}{h} u_\tau$$

where

$$u_\tau = \left(\|\boldsymbol{v}\|^2 + \frac{3}{2}c^2 + c\sqrt{16\|\boldsymbol{v}\|^2 + c^2} \right)^{1/2}$$

h is the characteristic mesh size, μ is the viscosity, κ is the thermal conductivity, c_v is the specific heat at constant volume and c is the acoustic speed. It is worth noting that the definition of the Courant number resembles the definition employed in [17] but is generalised to higher order approximation by introducing the order of approximation p.

All examples involve geometries with curved boundaries and the high–order computations employ high–order isoparametric curved meshes.

5.1. Subsonic inviscid flow around a circular cylinder

The first example involves subsonic flow around a circular cylinder, at a free-stream Mach number $M_\infty = 0.3$. This is a classical test case for inviscid flow solvers and is often used to quantify the amount of dissipation introduced by a numerical scheme. To demonstrate the mesh convergence of the SUPG formulation, using different degrees of approximation, five O–meshes are considered. The meshes are obtained from a successive refinement of an initial triangular mesh. A detail of views of these meshes near the cylinder is shown in Figure 3. For this simulation, the far field boundary is in the form of a circle and is located at a distance of 20 diameters from the cylinder. The entropy error, which is defined as

$$\epsilon_{\texttt{ent}} = \frac{P}{P_\infty}\left(\frac{\rho_\infty}{\rho}\right)^\gamma - 1$$

will be used to quantify the accuracy of the computed results.

Figure 4 demonstrates how the linear, quadratic and cubic approximations converge as the mesh is refined. The $\mathscr{L}^2$–norm of the entropy error

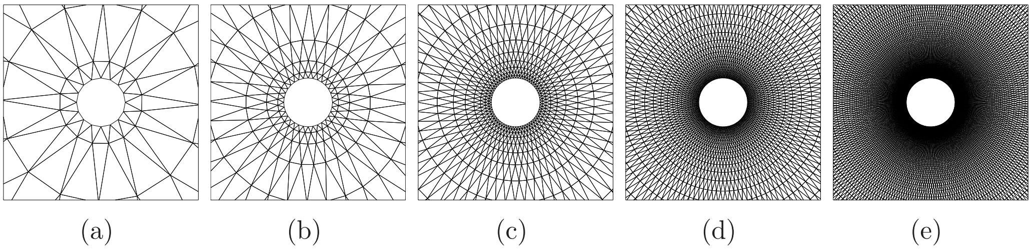


Figure 3: Detail of five O–meshes used for the analysis of flow over a circular cylinder: (a) mesh 1, 128 elements; (b) mesh 2, 512 elements; (c) mesh 3, 2 048 elements; (d) mesh 4, 8 192 elements; (e) mesh 5, 32 768 elements.

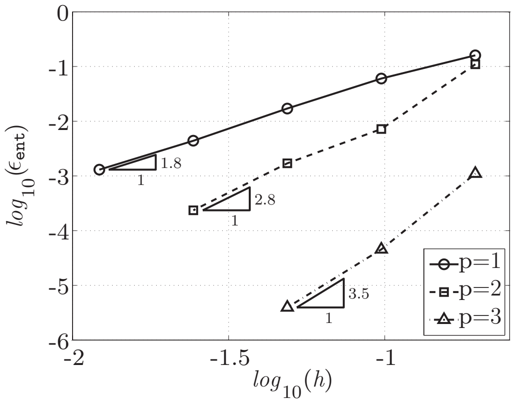


Figure 4: Subsonic inviscid flow around a circular cylinder: convergence of the entropy error, $\epsilon_{\texttt{ent}}$, with characteristic mesh size, h.

over the cylinder boundary is presented as a function of the characteristic mesh size, h. For a given mesh, the value of h is obtained as the minimum inscribed circle radius, taken over all the elements. For pure advection, the optimal rate of convergence for the SUPG formulation is $\text{p} + 1/2$ [11]. The rate of convergence here is slightly better than predicted for the linear and quadratic approximations, but is as predicted for the cubic approximation. It can be observed that, for the linear approximation, the finest mesh provides the same accuracy as is obtained on the coarsest mesh using the cubic approximation, i.e. the accuracy achieved using linear elements can also be achieved by using cubic elements, but with only 4% of the degrees of freedom.

This example has been used by several authors to study the accuracy of the geometrical representation required when the Euler equations are solved using the discontinuous Galerkin (DG) method [21, 22]. With a piecewise

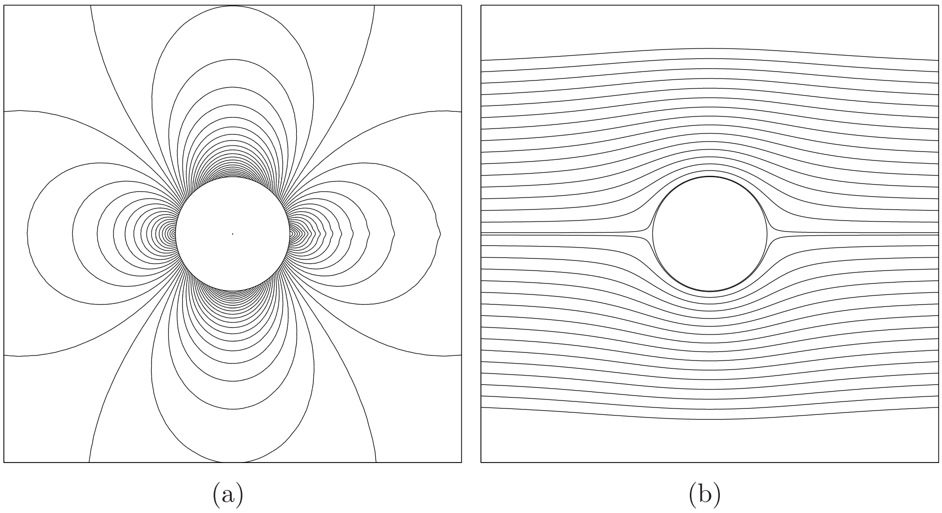


Figure 5: Subsonic inviscid flow over a circular cylinder: (a) Mach number isolines and (b) streamlines computed on the mesh of Figure 3e with linear elements.

linear approximation of the geometry, a non–physical entropy production is observed behind the cylinder in the DG solution, preventing the convergence to the correct symmetric steady state solution. For the continuous stabilised finite element formulation considered here, the steady state solution is obtained even with a linear approximation of the geometry, indicating that the formulation is less sensitive to inaccuracies in the geometry definition compared to the DG method. Figure 5 shows the distribution of the Mach number isolines and the streamlines, computed on the finest mesh with linear elements, and the symmetry of the numerical solution is clearly apparent.

5.2. Subsonic inviscid flow over a NACA0012 profile

The next example involves the simulation of subsonic inviscid flow over a NACA0012 aerofoil. The free–stream Mach number is $M_\infty = 0.63$ and the angle of attack is $\alpha = 2^o$. This example is used to compare the performance of the linear and the higher–order approximations for a more complicated inviscid problem with a smooth solution.

Six triangular meshes are employed. A non–uniform mesh refinement is adopted in the vicinity of the aerofoil and towards the leading and trailing edges, to capture the complex flow features in these regions. A detail of the first five of these meshes is shown in Figure 6.

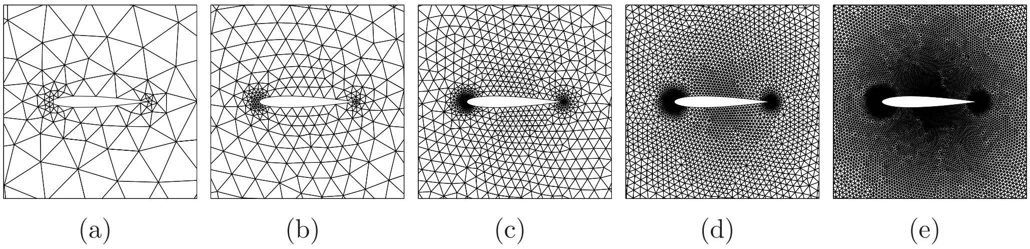


Figure 6: Detail of the meshes employed to simulate inviscid subsonic flow over a NACA0012 aerofoil: (a) mesh 1, 607 elements; (b) mesh 2, 1 068 elements; (c) mesh 3, 2 742 elements; (d) mesh 4, 9 004 elements; (e) mesh 5, 33 072 elements. Mesh 6, 128 309 elements is not shown.

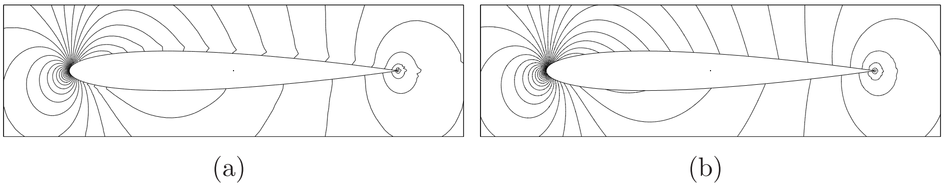


Figure 7: Subsonic inviscid flow over a NACA0012 aerofoil: detail of the Mach number isolines computed with (a) linear elements and the mesh of Figure 6e and (b) cubic elements and the mesh of Figure 6b.

Figure 7a shows a detail of the Mach number isolines, near the aerofoil, computed with linear elements on the mesh of Figure 6e. The equivalent results for a computation with cubic elements on the mesh of Figure 6b are shown in Figure 7b. Despite the very fine mesh employed for the linear computation, the polygonal approximation of the boundary generates an entropy layer that pollutes the solution in the vicinity of the aerofoil, as seen in Figure 7a. As might be expected for a problem with a smooth solution, significantly more accurate results are obtained with very coarse meshes and higher–order approximations, as can be observed in Figure 7b.

The performance of the linear and the higher–order approximations has been analysed in terms of the CPU time required, and the number of degrees of freedom used, to obtain the lift and drag coefficients to a desired accuracy. Figure 8 shows the CPU time required to converge these coefficients, as the mesh is refined, with different orders of approximation. In this figure, each line represents an order of approximation and each symbol represents a level of mesh refinement. The CPU time is scaled with respect to the time required for the computation with linear elements on the finest mesh. The

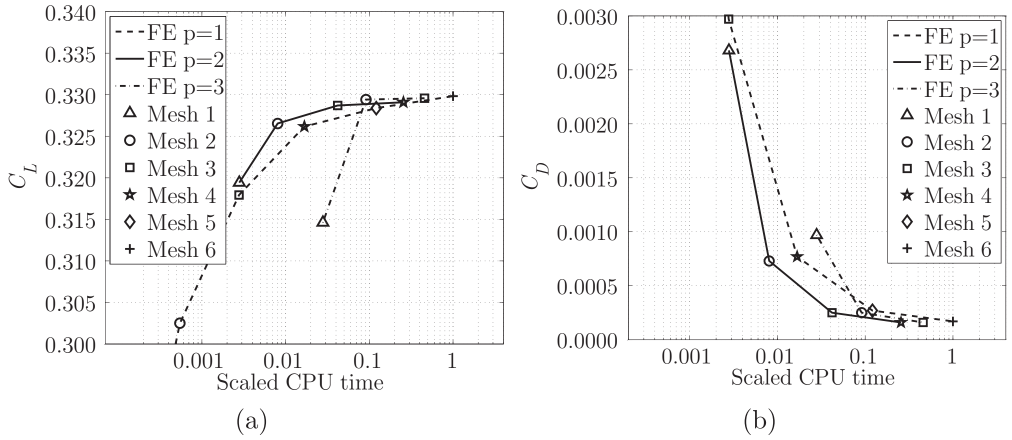


Figure 8: Subsonic inviscid flow over a NACA0012 aerofoil: convergence of (a) lift and (b) drag coefficients, with mesh refinement, as a function of the CPU time.

convergence of the lift coefficient, computed using different orders of approximation, is demonstrated in Figure 8a. With linear elements, the computed value on the fifth and the sixth meshes is 0.328 and 0.330 respectively, which is a variation of two lift counts. With quadratic elements, mesh convergence is achieved on the third and fourth meshes, where the computed lift coefficient is 0.329. Finally, with cubic elements, mesh convergence is achieved on the second and third meshes, where the computed lift coefficient is again 0.329. These results show that the computation with the cubic elements on the mesh of Figure 6b requires 11 times less CPU time than the computation with the linear elements on the finest mesh. The computation with quadratic elements, on the mesh of Figure 6c, requires 24 times less CPU time than the computation with the linear elements on the finest mesh. It is worth remarking that mesh convergence of the lift coefficient has not been attained with linear elements on the finest mesh, so a correct evaluation of the reduction in the CPU time achieved using the higher–order elements would be even more significant.

Figure 8b shows that higher–order elements are also advantageous for achieving mesh–converged values for the drag coefficient. With linear elements, the difference between the computed drag coefficient on the fifth and the sixth meshes is one drag count. With quadratic elements, the same difference is already achieved with the third and the fourth meshes and, with cubic elements, this variation is achieved on the second and third meshes.

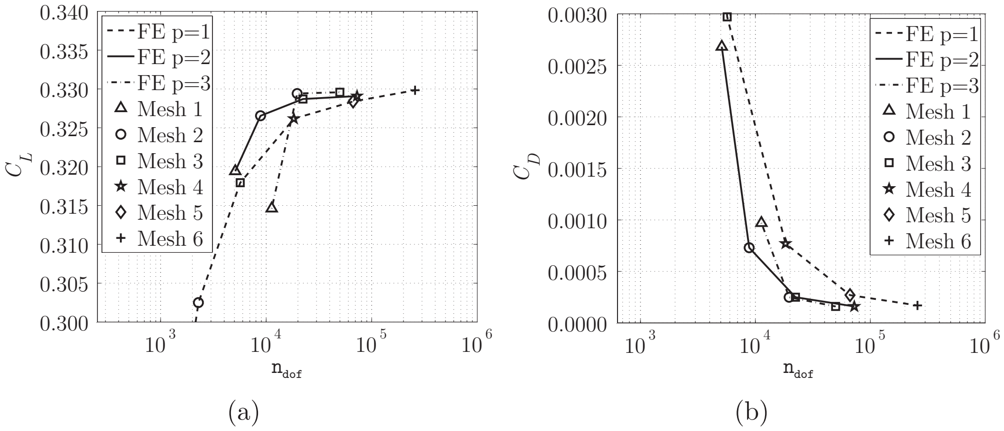
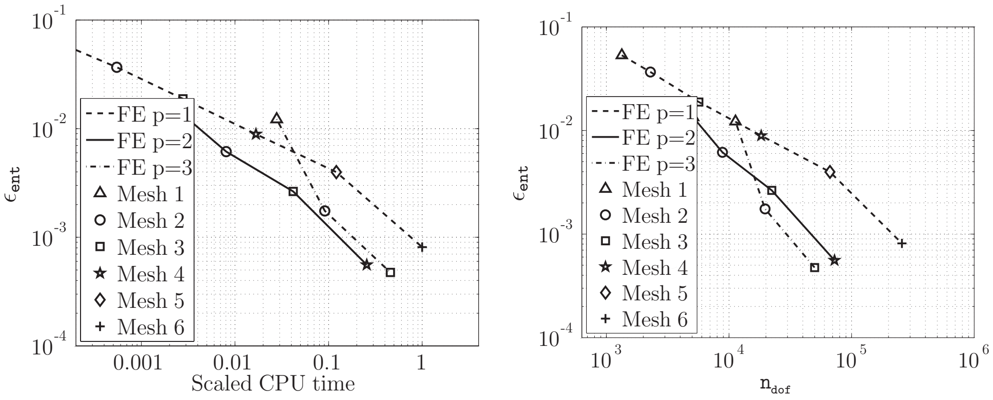


Figure 9: Subsonic inviscid flow over a NACA0012 aerofoil: convergence of (a) lift and (b) drag coefficients, with mesh refinement, as a function of the number of degrees of freedom, $\mathtt{n_{dof}}$.

Compared to linear elements, quadratic elements produce results of a similar quality using only a quarter of the CPU time. For cubic elements, equivalent results are obtained using about one third of the CPU time required with linear elements.

Figure 9 compares the performance of higher–order elements in terms of the number of degrees of freedom. Again, for a problem with a smooth solution, the use of higher–order elements results in an important saving in terms of the number of degrees of freedom required. Converged lift and drag coefficients are obtained with cubic elements on the third mesh, while the finest mesh is required to demonstrate mesh convergence of the drag coefficient with linear elements. This means that, with a cubic approximation, the number of degrees of freedom is reduced by more than a factor of five. Converged lift and drag coefficients are obtained with quadratic elements on the fourth mesh, using less CPU time than is required with cubic elements on the third mesh, as shown in Figure 8b. The number of degrees of freedom is slightly lower when cubic elements are employed, as shown in Figure 9b.

The entropy error can again be used to quantify the accuracy of the computations. Figure 10 shows how the entropy error on the aerofoil surface converges, as a function of the CPU time and with the number of the degrees of freedom employed. Again it is apparent that to obtain a desired accuracy, higher–order elements are not only able to reduce significantly the number



(a) (b)

Figure 10: Subsonic inviscid flow over a NACA0012 aerofoil: convergence of the entropy error as a function of (a) the CPU time and (b) the number of degrees of freedom, $\texttt{n}_{\texttt{dof}}$.

of degrees of freedom that must be used, but they are also more efficient in terms of the CPU time required.

5.3. Transonic inviscid flow over a NACA0012 profile

The final inviscid example involves transonic flow over a NACA0012 aerofoil. The free–stream Mach number is $M_\infty = 0.8$ and the angle of attack is $\alpha = 1.25^o$. This example is used to illustrate the performance of the linear and higher–order methods when the discontinuity capturing operator is employed to compute solutions involving shocks. Five triangular meshes are used and a detail of the meshes near the aerofoil is shown in Figure 11. As in the previous example, these meshes are refined in the vicinity of both the leading and the trailing edges, but extra mesh refinement has not been employed to facilitate shock capturing.

Figure 12 shows the pressure isolines computed with linear elements on the mesh of Figure 11e and with cubic elements on the mesh of Figure 11c. Both the strong and weak shocks are better resolved in the linear element computation, because the element size is smaller. In fact, the finest mesh has 374 nodes on the aerofoil surface, while this number reduces to 282 for the third mesh. Although the shock resolution is better, the overall quality of the solution obtained using cubic elements on the coarse mesh is good, with no oscillations polluting the solution in the vicinity of the shocks. Figure 13 shows the corresponding computed distributions of the pressure coefficient

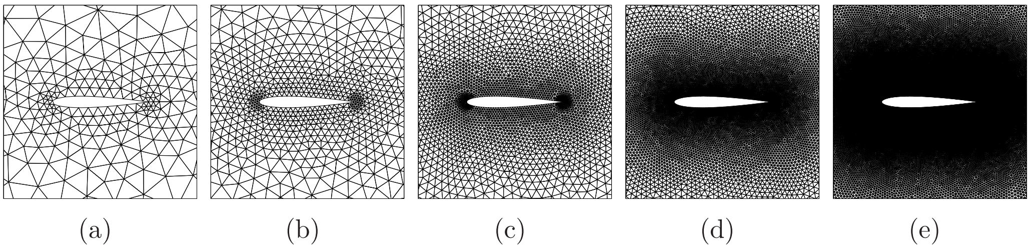


Figure 11: Detail of five computational meshes used to simulate inviscid transonic flow over a NACA0012 aerofoil: (a) mesh 1, 875 elements; (b) mesh 2, 2 041 elements; (c) mesh 3, 6 390 elements; (d) mesh 4, 23 139 elements; (e) mesh 5, 88 626 elements.

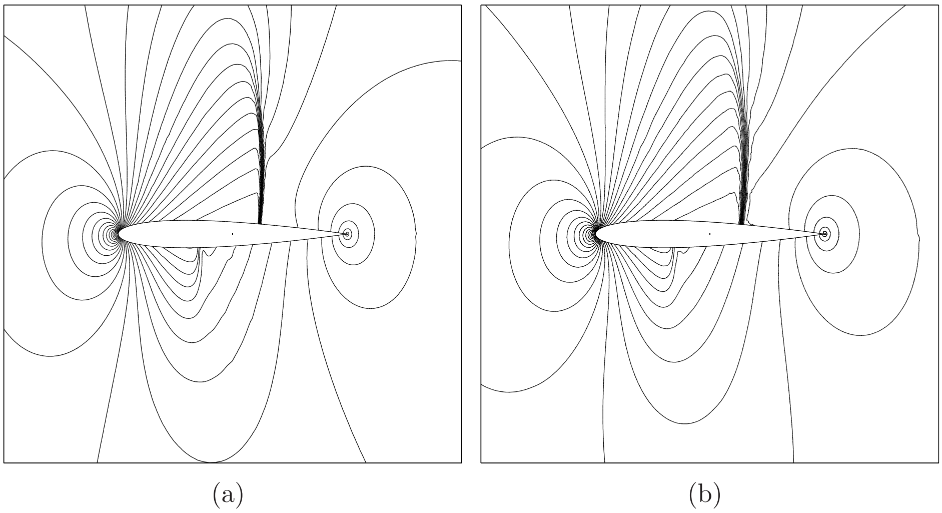
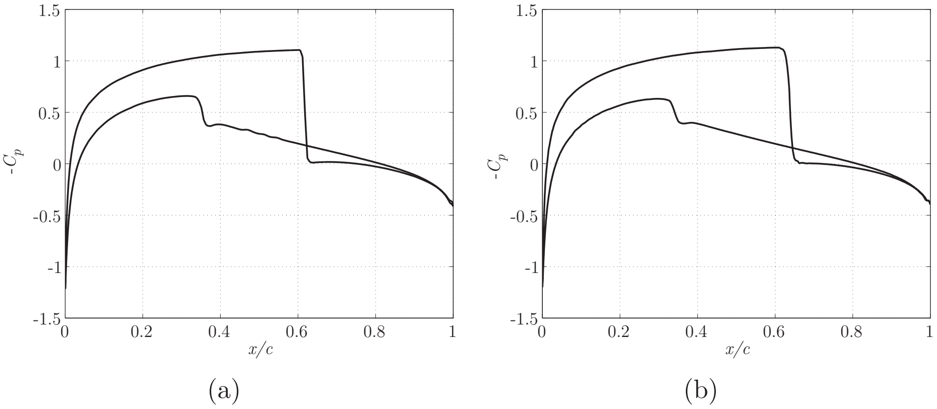


Figure 12: Transonic inviscid flow over a NACA0012 aerofoil: pressure isolines for (a) linear elements on the fifth mesh and (b) cubic elements on the third mesh.



(a)

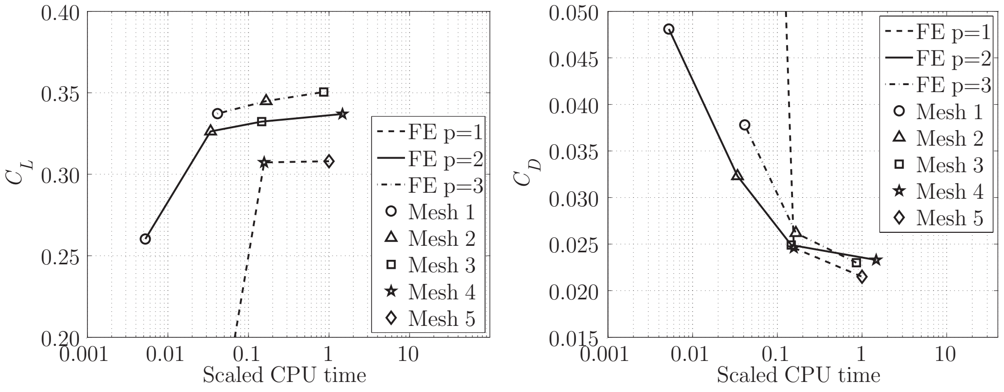
(b)

Figure 13: Transonic inviscid flow over a NACA0012 aerofoil: pressure coefficient obtained with (a) linear elements on the fifth mesh and (b) cubic elements on the third mesh.

on the aerofoil surface. Small oscillations are present behind the weak shock on the undersurface of the aerofoil when using linear elements. With cubic elements, a well resolved pressure coefficient distribution is obtained, with no overshoots or undershoots in the vicinity of the shocks.

Figure 14 demonstrates how the computed lift and drag coefficients converge, with the CPU time, as the mesh is refined and using different orders of approximation. In this figure, each line represents an order of approximation and each symbol represents a level of mesh refinement. Experimental values for this example have been reported [23] and, assuming a tolerance of five lift/drag counts, converged lift and drag coefficients should lie within the ranges of $(0.342, 0.352)$ and $(0.0217, 0.0227)$ respectively.

For the lift coefficient, Figure 14a indicates that an acceptable value is obtained with cubic elements and the mesh of Figure 11b. For this case, the computed lift coefficient is 0.345, which is just two lift counts lower than the experimental value reported in [23]. However, with linear elements, even on the finest mesh of Figure 11e, the computed lift coefficient is 0.308, which is 37 lift counts lower than the experimental value and outside the acceptable range. Quadratic elements are significantly better than linear elements but, even on the mesh of Figure 11d, they produce a computed lift of 0.337, which is 10 lift counts lower than the experimental value and not within the acceptable range. The lift computed with cubic elements on the mesh of Figure 11b is five times more accurate than the result obtained with quadratic elements and the mesh of Figure 11d and the corresponding CPU



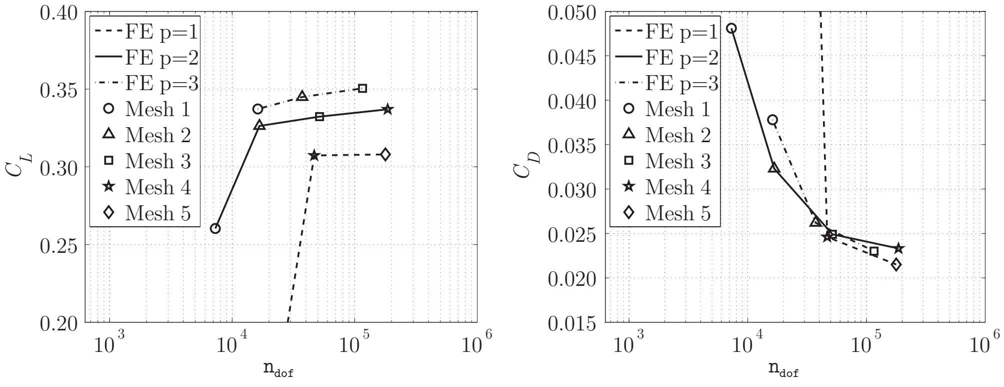
(a) (b)

Figure 14: Transonic inviscid flow over a NACA0012 aerofoil: convergence of (a) the lift and (b) the drag coefficients, with mesh refinement, as a function of the CPU time.

time is reduced by a factor of nine. The lift computed with cubic elements on the mesh of Figure 11b is 13 times more accurate than the result of the computation with linear elements on the mesh of Figure 11e. In this case, the corresponding CPU time is reduced by a factor of six. Although further mesh refinement could be attempted with either the linear or the quadratic elements, the required computational effort would be much greater than that required with cubic elements.

Similar conclusions are reached when considering the behaviour of the computed drag coefficient, shown in Figure 14b. For example, with cubic elements, the computed drag coefficient is seven drag counts higher than the experimental value, while the computed drag coefficient is seven drag counts lower than the experimental value when using linear elements. In terms of CPU time, a 15% saving is obtained for the same accuracy when using cubic elements compared with linear elements. The use of quadratic elements does not appear to present any advantage in this case.

Figure 15 shows the convergence of both the lift and drag coefficients with the number of degrees of freedom, as the mesh is refined, and using different orders of approximation. The results demonstrate the improvement in the estimation of the lift and drag coefficients resulting from the use of cubic elements. With cubic elements, the lift coefficient computed on the mesh of Figure 11b is five times more accurate than the value computed with quadratic elements and the mesh of Figure 11d. The number of degrees



(a) (b)

Figure 15: Transonic inviscid flow over a NACA0012 aerofoil: convergence of (a) the lift and (b) the drag coefficients, with mesh refinement, as a function of the number of degrees of freedom, $\mathtt{n_{dof}}$.

of freedom required with the cubic approximation is five times lower. The lift computed with cubic elements and the mesh of Figure 11b is 13 times more accurate than the result obtained from the computation with linear elements and the mesh of Figure 11e, with the cubic approximation requiring five times fewer degrees of freedom.

For the drag coefficient, linear and cubic elements provide the same level of accuracy, but with cubic elements requiring 35% fewer degrees of freedom. Again, the use of the quadratic approximation does not appear to present any advantage in this case.

The difference between the converged lift when using linear and higher order approximation is attributed to the difference in the geometric representation of the curved boundary. Whit linear elements, a polygonal approximation of the boundary introduces corners that are expected to generate much more entropy compared to the better approximation of the boundary that is employed with higher order isoparametric elements, see for instance the comparison of spurious entropy production for the previous example in Figure 10.

5.4. Subsonic viscous flow over a circular cylinder

The next example involves laminar viscous flow over a circular cylinder. The Reynolds number, based on the diameter of the cylinder, is Re=30, the free–stream Mach number is $M_\infty = 0.1$ and the angle of attack is $\alpha = 0^o$.

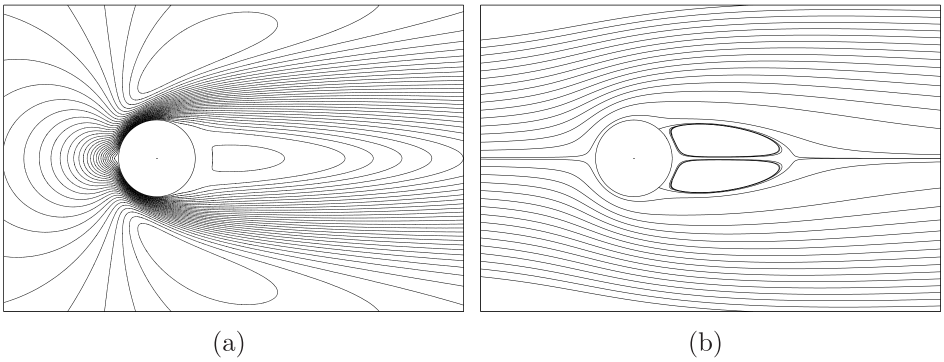
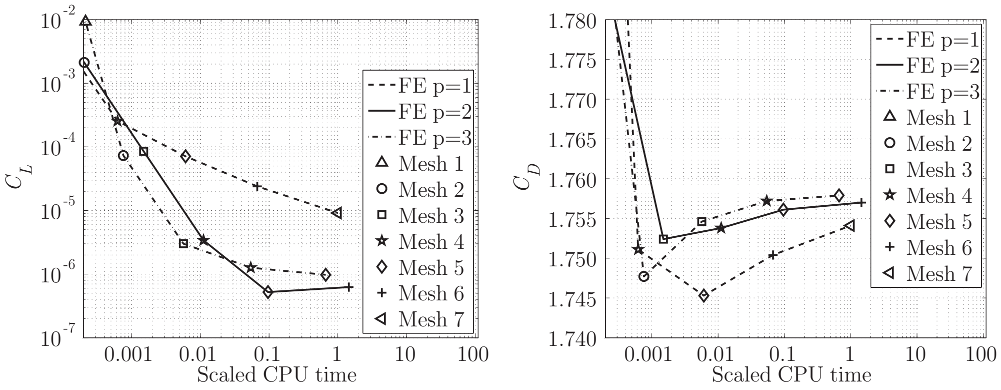


Figure 16: Subsonic viscous flow over a circular cylinder: (a) Mach number isolines and (b) streamlines computed with cubic elements on the mesh of Figure 3d.

This example is used to evaluate the performance of the SUPG formulation for a simple viscous test case and to investigate its behaviour when the flow approaches the incompressible limit.

The isotropic meshes employed earlier for the computation of inviscid flow over a circular cylinder are again employed, with no special refinement added to aid the capture of the stationary regions of recirculating flow that are known to appear immediately behind the cylinder. Two additional meshes, with 131 072 and 524 288 elements respectively, are considered from a successive refinement of the mesh of Figure 3e. Figure 16 shows the Mach number isolines and the streamlines computed with cubic elements on the mesh of Figure 3d. Figures 17 and 18 show how the lift and drag coefficients converge, with the CPU time and with the number of degrees of freedom respectively, as the mesh is refined, using different orders of approximation. The flow field symmetry means that the lift force on the cylinder should be zero, so that the computed lift force can be employed as a measure of the error in the computation. The lift coefficient is rapidly converged to an accuracy of one lift count, as shown in Figures 17a and 18a. No important differences are observed in computational performance between the linear and the higher–order approximations. There is a small saving in terms of the number of degrees of freedom obtained by using the higher–order approximations. For higher accuracy, quadratic and cubic elements outperform linear elements, not only in term of number of degrees of freedom employed, but also in terms of the CPU time required.

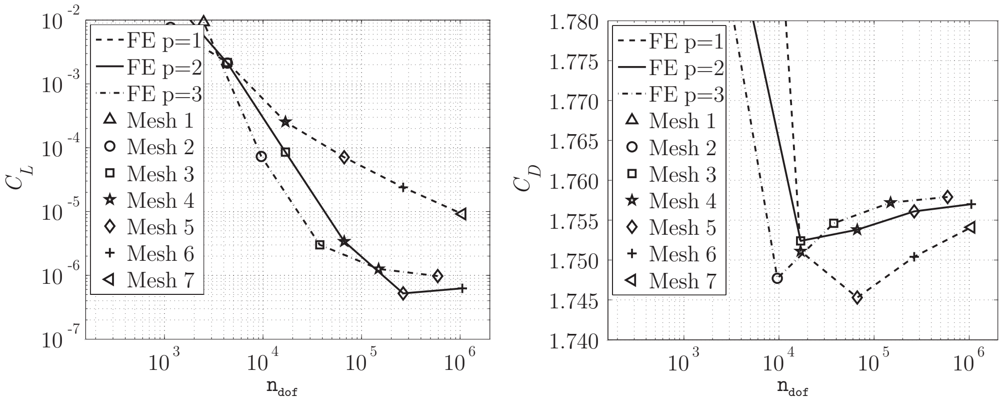
The drag coefficient is more difficult to converge and the conclusions are



(a)

(b)

Figure 17: Subsonic viscous flow over a circular cylinder: convergence of (a) the lift and (b) the drag coefficients, with mesh refinement, as a function of the CPU time.



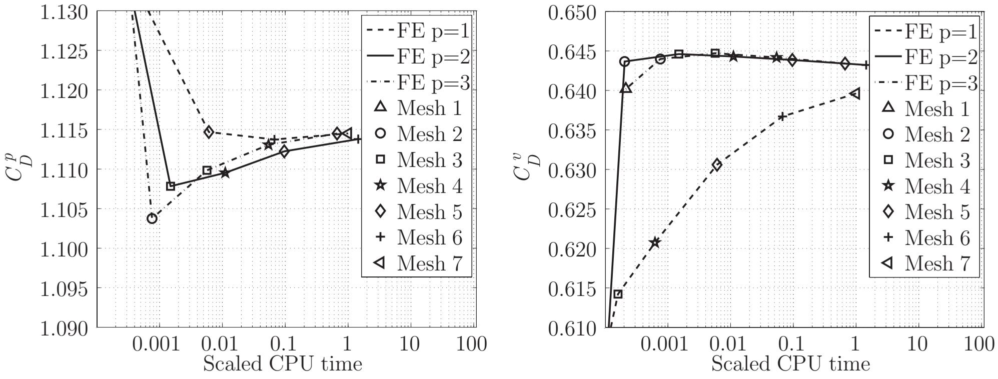
(a)

(b)

Figure 18: Subsonic viscous flow over a circular cylinder: convergence of (a) the lift and (b) the drag coefficients, with mesh refinement, as a function of the number of degrees of freedom, $\mathtt{n_{dof}}$.

different. With linear elements, the computed drag coefficient on the sixth mesh is 1.7504, which differs by 37 drag counts from the value of 1.7541 obtained on the seventh mesh. In contrast, with the quadratic elements, the computed drag coefficient on the mesh of Figure 3e is 1.7561, which differs by only 9 drag counts from the value of 1.7570 obtained on the sixth mesh. With cubic elements, the computed drag coefficient on the mesh of Figure 3d is 1.7572, which is only 7 drag counts different from the result obtained with the mesh of Figure 3e. The accuracy obtained on the finest mesh with the linear approximation is the same as that obtained with quadratic elements on the fourth mesh, i.e. the same accuracy is obtained with a reduction of a factor of 90 in the CPU time required, as shown in Figure 17b. At the same time, the number of degrees of freedom employed is reduced by a factor of 15, as illustrated in Figure 18b. Cubic elements outperform both linear and quadratic approximations, with the accuracy obtained on the finest mesh with the linear approximation being reached on the fourth mesh with cubic elements. In this case, the CPU time is reduced by a factor of 180, as shown in Figure 17b, and the number of degrees of freedom is reduced by a factor of 28, as shown in Figure 18b.

To illustrate the differences between the linear and the higher–order approximations, Figure 19 shows how the pressure and the viscous drag coefficient components converge, with the CPU time, as the mesh is refined. No important differences in performance are observed for the pressure drag coefficient component, C_D^p, as shown in Figure 19a. For example, a computed pressure drag coefficient of 1.1145 is obtained with the linear elements on the finest mesh and with the cubic elements on the fifth mesh, with no remarkable difference in the CPU time. In contrast, the convergence of the viscous drag coefficient, C_D^v, shows dramatic differences between the results obtained with the linear and the higher–order approximations. With linear elements, the variation of the computed viscous drag coefficient between the sixth and the finest mesh is 30 drag counts, whereas with cubic elements the variation between the fourth and fifth meshes is only 7 drag counts. It is also worth emphasising that, with quadratic and cubic elements, the second mesh provides an acceptable value for the viscous drag coefficient component. To illustrate the differences between linear and higher–order approximations, Figure 20 shows the pressure coefficient and the skin friction distribution on the upper surface of the cylinder. The results for a computation with linear elements on the sixth mesh and for a computation with cubic elements on the fourth mesh are displayed. A perfect match is obtained for the pressure



(a) (b)

Figure 19: Subsonic viscous flow over a circular cylinder: convergence of (a) the pressure drag and (b) the viscous drag coefficients, with mesh refinement, as a function of the CPU time.

coefficient, whereas slight differences are observed between the two skin friction distributions. This fact corroborates the conclusions in [24], where it was reported that the viscous forces are usually better resolved by using a quadratic approximation compared to a linear approximation.

It is worth mentioning that the global reconstruction of the derivative of the viscous fluxes does not have a big impact int the computed aerodynamic quantities with linear elements. For this example, a difference of just one drag count has been observed when the reconstruction is used compared to the computation performed by assuming that the second derivatives are zero. In addition, the computation involving the reconstructed derivatives employs approximately 5% more CPU time.

5.5. Transonic viscous flow over a NACA0012 profile

The final example involves laminar viscous flow over a NACA0012 aerofoil profile. The Reynolds number, based on the chord length of the aerofoil, is Re=500, the free–stream Mach number is $M_\infty = 0.8$ and the angle of attack is $\alpha = 10^{\text{o}}$ [25, 26]. This example tests the performance of linear and high–order approximations for a more complex viscous example and evaluates the competitiveness of the stabilised finite element approach with respect to the computational requirements of a well–established second order finite volume procedure [27, 28, 29]. The computations are again performed on the meshes shown in Figure 11.

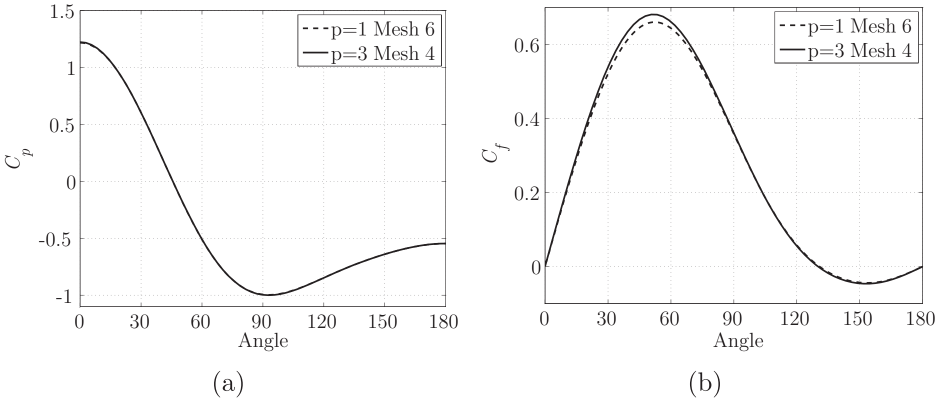
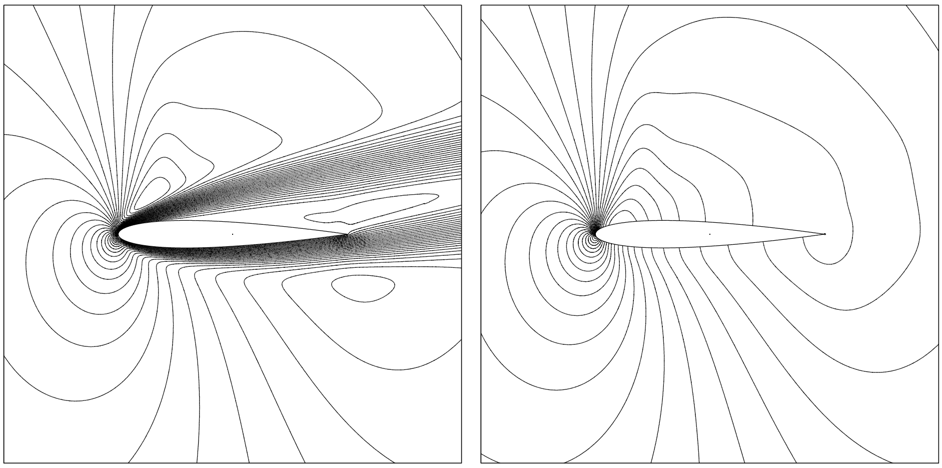


Figure 20: Subsonic viscous flow over a circular cylinder: (a) the pressure coefficient and (b) the skin friction computed with linear elements on the sixth mesh and with cubic elements on the fourth mesh.

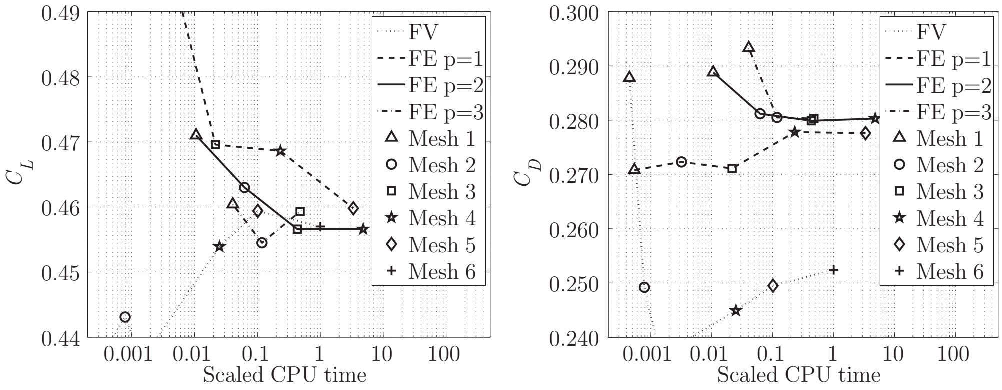
Figure 21 shows the Mach number and the pressure isolines computed with cubic elements on the mesh of Figure 11c. The ability of the higher–order approximations to accurately capture the flow features using a coarse mesh can be clearly observed. Figures 22 and 23 show the convergence of the lift and drag coefficients with the CPU time and the number of degrees of freedom respectively, as the mesh is refined, using different orders of approximation. It is apparent, in Figures 22a and 23a, that the high angle of attack makes the lift coefficient more difficult to converge than in the previous examples. Using linear elements, the difference in the lift coefficient computed on the fourth and the fifth meshes is nine lift counts. With quadratic elements, the value of the lift coefficient is converged on the third mesh, with no variation obtained on the finer meshes. With cubic elements, the lift coefficient computed on the second and the third meshes varies by 48 lift counts. The finite volume code shows a similar performance, with the difference of 24 lift counts between the values of the lift coefficient computed on the fifth and sixth meshes.

The convergence results for the drag coefficient show different conclusions, as shown in Figures 22b and 23b. With linear elements, the difference between the drag coefficient computed on the fourth and fifth meshes is only two drag counts. The difference between the drag coefficients computed on the third and fourth meshes with quadratic elements is four drag counts. With cubic elements, there is a difference of only two drag counts between



(a) (b)

Figure 21: Transonic viscous flow over a NACA0012 aerofoil: (a) Mach number isolines and (b) pressure isolines computed with cubic elements on the mesh of Figure 11c.



(a) (b)

Figure 22: Transonic viscous flow over a NACA0012 aerofoil: convergence of (a) the lift and (b) the drag coefficients, with mesh refinement, as a function of the CPU time.

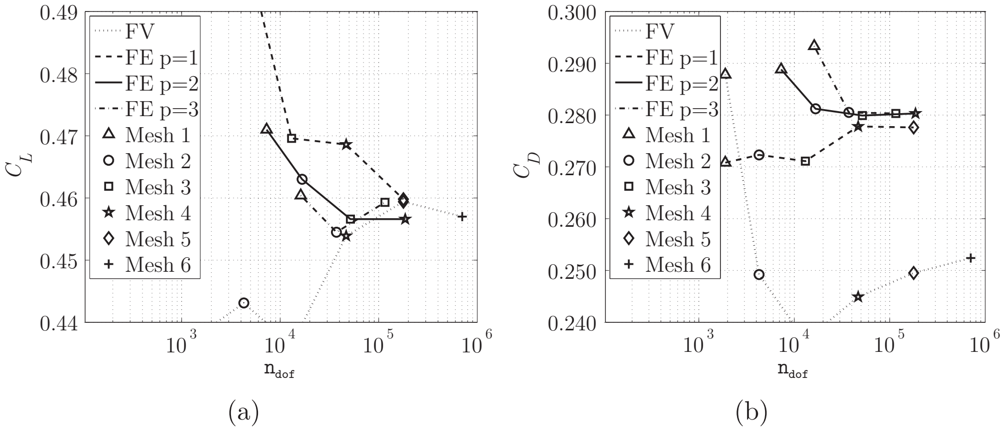


Figure 23: Transonic viscous flow over a NACA0012 aerofoil: convergence of (a) the lift and (b) the drag coefficients, with mesh refinement, as a function of the number of degrees of freedom, $\mathtt{n_{dof}}$.

the drag coefficients computed on the second and third meshes. The finite volume code shows a poor performance in converging this aerodynamic force, giving a difference of more than 70 drag counts between the drag coefficients computed on the fifth and sixth meshes. For this example, it is apparent that, not only is the number of degrees of freedom significantly reduced by using higher–order finite elements, but so also is the CPU time. The advantages offered by adopting higher–order methods, in place of a classical second order finite volume solution procedure, are clearly apparent. It is worth remarking that the finite volume solver requires further refinement to converge this aerodynamic quantity, but since the solution obtained with the finite volume solver in the sixth mesh already requires more CPU time and degrees of freedom than the converged solution using linear or higher order stabilised finite elements, this clearly demonstrates that finite volume methods are not competitive.

Although the lift and drag coefficients computed here, with both the linear and the higher–order stabilised finite element method, and also with the second order finite volume code approach, all lie within the range of published values for this example [30], there is a discrepancy between the linear and the higher–order methods in the converged values of the drag coefficient. The drag coefficient is converged on the fourth mesh, when quadratic elements are employed, and on the third mesh with cubic elements, whereas the converged value obtained with linear elements differs by 27 drag counts. The

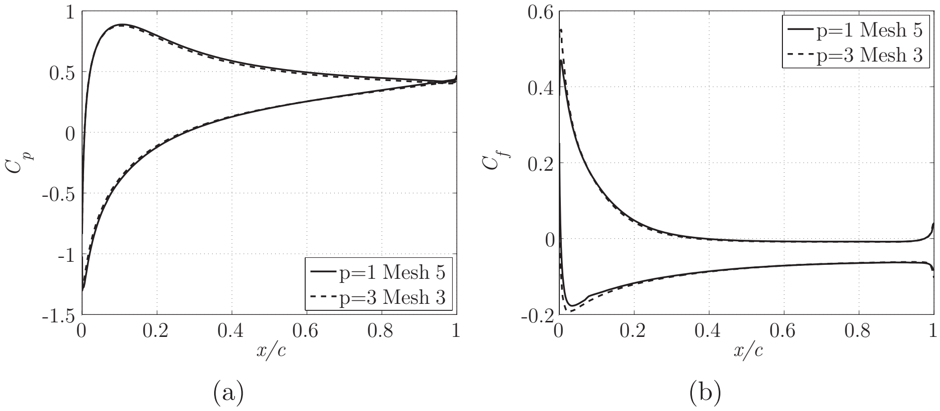


Figure 24: Transonic viscous flow over a NACA0012 aerofoil: (a) the pressure coefficient and (b) the skin friction on the aerofoil surface computed with linear elements on the fifth mesh and with cubic elements on the third mesh.

reason for this discrepancy is again the use of meshes with straight–sides for linear approximations and curved meshes for higher order approximations. As it has been clearly shown when simulating inviscid flows, the use of high–order curved elements substantially reduces the spurious entropy production near curved walls when compared to standard linear elements, improving the quality of the solution in the vicinity of curved boundaries and, therefore, improving the accuracy of the computed aerodynamic forces. To illustrate the reasons for this difference, Figure 24 shows the computed pressure coefficient and skin friction distribution on the aerofoil surface. The displayed distributions are obtained with linear elements on the mesh of Figure 11e and with cubic elements on the mesh of Figure 11c. Again, for this example, there are no significant differences between the computed pressure distributions, but discrepancies are apparent in the skin friction distributions near the leading edge of the aerofoil. Again, this fact corroborates the conclusions obtained in the previous example and the conclusions in [24], where it was reported that the viscous forces are better resolved by using higher order approximations.

6. Conclusions

The application of the stabilised SUPG formulation, combined with higher–order polynomial approximations in space and discontinuous approximations in time, has been considered. The resulting method is high–order accurate in space and provides an implicit unconditionally stable time marching algorithm.

Optimal convergence of the higher–order SUPG formulation has been illustrated in the case of a simple subsonic inviscid flow example. More complex two dimensional examples, involving inviscid and viscous subsonic and transonic compressible flows, have been employed to compare and discuss the performance of the linear and higher–order approximations. The relative performance of the linear and higher–order approaches has been compared in terms of the number of degrees of freedom and the CPU time required to converge lift and drag coefficients. The performance of the SUPG formulation has also been compared to that of a standard second order finite volume solution algorithm.

The presented results show the advantages of higher–order approximations in this context. They are able not only to reduce drastically the necessary number of degrees of freedom but, more importantly, to reduce the CPU time required to reach the desired accuracy. The possibilities offered by the higher–order stabilised finite element approach for industrial applications have been demonstrated by the results of the comparison with the performance of the second order finite volume method.

References

[1] N. Kroll, The ADIGMA project, in: N. Kroll, H. Bieler, H. Deconinck, V. Couaillier, H. van der Ven, K. Sørensen (Eds.), ADIGMA – A European initiative on the development of adaptive higher–order variational methods for aerospace applications, volume 113 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, Springer, 2010, pp. 1–9.

[2] B. Cockburn, Discontinuous Galerkin methods for computational fluid dynamics, in: E. Stein, R. de Borst, T. Hughes (Eds.), Encyclopedia of Computational Mechanics, volume 3, Wiley, New York, 2004, pp. 91–128.

[3] K. A. Sørensen, H. Bieler, Verification and assessment, in: N. Kroll, H. Bieler, H. Deconinck, V. Couaillier, H. van der Ven, K. Sørensen (Eds.), ADIGMA – A European initiative on the development of adaptive higher–order variational methods for aerospace applications, volume 113 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, Springer, 2010, pp. 465–482.

[4] A. N. Brooks, T. J. R. Hughes, Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations, Comput. Methods Appl. Mech. Eng. 32 (1982) 199–259.

[5] T. Hughes, L. Franca, G. Hulbert, A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least–squares method for advective–diffusive equations, Comput. Methods Appl. Mech. Eng. 73 (1989) 173–189.

[6] F. L. Chalot, Industrial aerodynamics, in: E. Stein, R. de Borst, T. Hughes (Eds.), Encyclopedia of Computational Mechanics, volume 3, Wiley, New York, 2004, pp. 407–458.

[7] D. Corson, R. Jaiman, F. Shakib, Industrial application of RANS modelling: capabilities and needs, Int. J. Comput. Fluid Dyn. 23 (2009) 337–347.

[8] T. Hughes, G. Scovazzi, T. E. Tezduyar, Stabilized methods for compressible flows, J. Sci. Comput. 43 (2010) 343–368.

[9] C. H. Whiting, K. E. Jansen, A stabilized finite element method for the incompressible Navier–Stokes equations using a hierarchical basis, Internat. J. Numer. Methods Fluids 35 (2001) 93–116.

[10] F. Chalot, P.-E. Normand, Higher–order stabilized finite elements in an industrial Navier–Stokes code, in: N. Kroll, H. Bieler, H. Deconinck, V. Couaillier, H. van der Ven, K. Sørensen (Eds.), ADIGMA – A European initiative on the development of adaptive higher–order variational methods for aerospace applications, volume 113 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, Springer, 2010, pp. 145–165.

[11] C. Johnson, U. Nävert, J. Pitkäranta, Finite element methods for linear hyperbolic problems, Comput. Methods Appl. Mech. Eng. 45 (1984) 285–312.

[12] K. Morgan, J. Peraire, Unstructured grid finite element methods for fluid mechanics, Rep. Prog. Phys. 61 (1998) 569–638.

[13] K. Jansen, S. Collis, C. Whiting, F. Shakib, A better consistency for low–order stabilized finite element methods, Comput. Methods Appl. Mech. Eng. 174 (1999) 153–170.

[14] J. Donea, A. Huerta, Finite Element Methods for Flow Problems, Wiley, 2005.

[15] T. Tezduyar, M. Senga, Stabilization and shock–capturing parameters in SUPG formulation of compressible flows, Comput. Methods Appl. Mech. Eng. 195 (2006) 1621–1632.

[16] B. S. Kirk, G. F. Carey, Development and validation of a SUPG finite element scheme for the compressible Navier–Stokes equations using a modified inviscid flux discretization, Internat. J. Numer. Methods Fluids 57 (2008) 265–293.

[17] F. Shakib, T. J. Hughes, Z. Johan, A new finite element formulation for computational fluid dynamics: X. The compressible Euler and Navier–Stokes equations, Comput. Methods Appl. Mech. Eng. 89 (1991) 141–219.

[18] P. G. Ciarlet, The Finite Element Method For Elliptic Problems, North–Holland, 1978.

[19] Y. Saad, M. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Comput. 7 (1986) 856–869.

[20] Y. Saad, Iterative methods for sparse linear systems, PWS Publishing, New York, 1996.

[21] F. Bassi, S. Rebay, High–order accurate discontinuous finite element solution of the 2D Euler equations, J. Comput. Phys. 138 (1997) 251–285.

[22] R. Sevilla, S. Fernández-Méndez, A. Huerta, NURBS–enhanced finite element method (NEFEM) for Euler equations, Internat. J. Numer. Methods Fluids 57 (2008) 1051–1069.

[23] J. J. Thibert, M. Granjacques, L. H. Ohman, NACA 0012 airfoil, AGARD Advisory Report 138, AGARD, 1979.

[24] C. H. Whiting, K. E. Jansen, S. Dey, Hierarchical basis for stabilized finite element methods for compressible flows, Comput. Methods Appl. Mech. Eng. 192 (2003) 5167 – 5185.

[25] D. J. Mavripilis, A. Jameson, L. Martinelli, Multigrid solution of the Navier–Stokes equations on triangular meshes, AIAA Journal 28 (1990) 1415–1425.

[26] G. May, B. Srinivasan, A. Jameson, An improved gas–kinetic BGK finite–volume method for three–dimensional transonic flow, J. Comput. Phys. 220 (2007) 856–878.

[27] K. A. Sørensen, A multigrid procedure for the solution of compressible fluid flows on unstructured hybrid meshes, Ph.D. thesis, University of Wales Swansea, 2002.

[28] O. Hassan, K. A. Sørensen, K. Morgan, N. P. Weatherill, A method for time accurate turbulent compressible fluid flow simulation with moving boundary components employing local remeshing, Internat. J. Numer. Methods Fluids 53 (2007) 1243–1266.

[29] B. Evans, O. Hassan, J. Jones, K. Morgan, L. Remaki, Computational fluid dynamics applied to the aerodynamic design of a land-based supersonic vehicle, Numer. Methods Partial Differential Equations 27 (2011) 141–159.

[30] M. O. Bristeau, R. Glowinski, J. Periaux, H. Viviand (Eds.), Proceedings of the GAMM workshop on the numerical simulation of compressible Navier–Stokes flows, held at INRIA, Sophia–Antipolis (France), December 1985, volume 18 of *Notes on Numerical Fluid Mechanics*, Vieweg–Verlag, 1986.