V European Conference on Computational Fluid Dynamics
ECCOMAS CFD 2010
J. C. F. Pereira and A. Sequeira (Eds)
Lisbon, Portugal,14-17 June 2010

SEMI-IMPLICIT DISCONTINUOUS GALERKIN FINITE ELEMENT METHOD FOR THE STEADY STATE COMPRESSIBLE FLOWS[1]

Vít Dolejší[1]

Charles University Prague,
Faculty of Mathematics and Physics,
Sokolovska 83, Prague,
Czech Republic,
e-mail: dolejsi@karlin.mff.cuni.cz

Key words: discontinuous Galerkin method, compressible Navier-Stokes equations, semi-implicit time discretization, linear algebraic systems, iterative solver, stopping criterion, adaptive choice of the time step

Abstract. *We deal with the numerical solution of the system of the compressible Navier-Stokes equations with the aid of the interior penalty Galerkin method. We employ a semi-implicit time discretization which leads to the solution of a sequence of linear algebraic systems. We present an efficient solution strategy of these systems with a focus on steady-state problems. Our approach is based on a simple adaptive technique for the choice of the time step and a relatively weak stopping criterion for iterative linear algebraic solvers. The presented numerical experiments show that the proposed strategy is efficient for steady-state problems using various grids, polynomial degrees of approximations and flow regimes. Particularly we show that the computational time for the solution of these systems is smaller than the computational time for their preparation. Finally, we apply this strategy with a minor modification to an unsteady flow.*

[1]This work is a part of the research project MSM 0021620839 financed by the Ministry of Education of the Czech Republic and was partly supported by the Grant No. 201/08/0012 of the Czech Grant Agency. Additionally, this work is a part of the research project ADIGMA, No. AST5-CT-2006-030719 financed within the 3rd Call of the 6th European Framework Programme.

1 Introduction

Our aim is to develop a sufficiently robust, efficient and accurate numerical scheme for the simulation of steady as well as unsteady viscous compressible flows. The *discontinuous Galerkin* (DG) methods have become very popular numerical technique for the solution of the compressible Navier-Stokes equations. DG space discretization uses (higher order) piecewise polynomial discontinuous approximation on arbitrary meshes. DG methods were employed in many papers for the discretization of compressible fluid flow problems, see, e.g., [1], [2], [3], [6], [8], [9], [12], [15], [16], [17], [18], and the references cited therein. The recent progress of the use of DG method for compressible flow simulations can be found in [19].

The time discretization can be carried out also by a discontinuous approximation (e.g., [18], [20]) but the most usual approach is an application of the method of lines. In this case, the Runge-Kutta methods are very popular for their simplicity and a high order of accuracy, see [1], [3], [4], [8]. Their drawback is a strong restriction to the size of the time step. To avoid this disadvantage, it is suitable to use an implicit time discretization, e.g., [2], [17]. However, a full implicit scheme leads to a necessity to solve a nonlinear system of algebraic equations at each time level which is rather expensive. Therefore, in [9], [11], we developed the *semi-implicit method* which is based on a suitable linearization of the inviscid and viscous fluxes. The linear terms are treated implicitly (by a multistep BDF formula) whereas the nonlinear ones by an explicit extrapolation which leads to a linear algebraic system at each time level. We called this approach the *backward difference formula – discontinuous Galerkin finite element* (BDF-DGFE) method.

The sequence of these linear algebraic systems should be solved by a suitable solver. It is advantageous to use an iterative method (e.g., GMRES method [22] with a suitable preconditioner) since the solution of the previous system can be used as an initial guess of the solution of the next system. Moreover, it is not necessary to solve the systems too precisely since they arise from a discretization of partial differential equations and therefore the systems already contain discretization errors. Numerical experiments presented in [9] showed that BDF-DGFE method is efficient for unsteady flow problems but its efficiency for steady-state flow regimes is very low. The solution of the linear algebraic systems consumes more than 95% of the total computational time.

We present a new solution strategy which significantly reduce the computational time for steady state flows in comparison with [9]. We were inspired by the idea of the *inexact Newton method* [5] where also a sequence of linear algebraic systems has to be solved. The key is to define a relatively weak stopping criterion which guarantees a convergence to the steady state solution but requires only a few GMRES steps at each time level.

Although we focus in this paper on steady-state flows, we employ the "unsteady" formulation since our aim is to solve also unsteady flows with the same method. This approach is practical in situations when it is not known a priori if the flow regime is steady or unsteady. Therefore, we develop a simple technique for the adaptive choice of the size

of the time step. In this paper, we consider only one step BDF method (= backward Euler method) for simplicity. An extension to n-step BDF ($n \geq 2$) is straightforward.

The content of the rest of the paper is the following. In Section 2, we introduce the system of the compressible Navier-Stokes equations. In Section 3, we recall the BDF-DGFE discretization of the Navier-Stokes equations from [9]. In Section 4, we discuss numerical solution of the arising linear algebraic systems. Particularly, we deal with the choice of preconditioner of the GMRES method, stopping criteria and the size of the time step. Section 5 contains a set of numerical experiments demonstrating the efficiency and accuracy of the proposed strategy and also its robustness. The concluding remarks are given in Section 6.

2 Compressible flow problem

Let $\Omega \subset I\!R^d,\ d = 2, 3$, be a bounded domain with a Lipschitz piecewise polynomial boundary and $T > 0$. We set $Q_T = \Omega \times (0, T)$ and by $\partial\Omega$ denote the boundary of Ω which consists of several disjoint parts. We distinguish inlet $\partial\Omega_i$, outlet $\partial\Omega_o$ and impermeable walls $\partial\Omega_w$, i.e. $\partial\Omega = \partial\Omega_i \cup \partial\Omega_o \cup \partial\Omega_w$. The system of the Navier-Stokes equations describing a motion of non-stationary viscous compressible flow can be written in the dimensionless form

$$\frac{\partial \boldsymbol{w}}{\partial t} + \sum_{s=1}^{d} \frac{\partial \boldsymbol{f}_s(\boldsymbol{w})}{\partial x_s} = \sum_{s=1}^{d} \frac{\partial}{\partial x_s} \left(\sum_{k=1}^{d} \boldsymbol{K}_{sk}(\boldsymbol{w}) \frac{\partial \boldsymbol{w}}{\partial x_k} \right) \quad \text{in } Q_T, \tag{1}$$

where

$$\begin{array}{ll} \boldsymbol{w} = \boldsymbol{w}(x,t) : Q_T \to I\!R^{d+2} & \text{the state vector,} \\ \boldsymbol{f}_s : I\!R^{d+2} \to I\!R^{d+2},\ s = 1, \dots, d, & \text{the inviscid fluxes,} \\ \boldsymbol{K}_{sk} : I\!R^{d+2} \to I\!R^{(d+2)\times(d+2)},\ s, k = 1, \dots, d, & \text{the viscous terms.} \end{array} \tag{2}$$

The forms of vectors $\boldsymbol{w}$, $\boldsymbol{f}_s$, $s = 1, \dots, d$, and matrices $\boldsymbol{K}_{sk}$, $s = 1, \dots, d$, can be found, e.g., in [9] or [14, Section 4.3]. We consider the Newtonian type of fluid accompanied by the state equation for perfect gas and the definition of the total energy. The system (1) is equipped with a suitable initial and boundary conditions, see [8], [9]. We only mention that we prescribe several Dirichlet boundary conditions on the inlet and impermeable walls and on the rest of boundary the Neumann boundary condition is used. The problem to solve the Navier-Stokes equations (1) equipped with the initial and boundary conditions will be denoted by (CFP) (compressible flow problem).

If we omit the time derivative term on the left-hand side of (1), we obtain the *stationary Navier-Stokes equations*. The problem to solve the stationary Navier-Stokes equations equipped with the same boundary conditions as in the non-stationary case will be denoted by (sCFP) (stationary compressible flow problem).

3 DGFE discretization

3.1 Triangulations

Let $\mathcal{T}_h$ $(h > 0)$ be a partition of the closure $\overline{\Omega}$ of the domain Ω into a finite number of closed d-dimensional elements K with mutually disjoint interiors. By $\mathcal{F}_h$ we denote the set of all open $(d-1)$-dimensional faces (open edges when $d = 2$ or open faces when $d = 3$) of all elements $K \in \mathcal{T}_h$. Further, the symbol $\mathcal{F}_h^I$ stands for the set of all $\Gamma \in \mathcal{F}_h$ that are contained in Ω (inner faces). Moreover, we introduce notations $\mathcal{F}_h^w$, $\mathcal{F}_h^i$ and $\mathcal{F}_h^o$ for the sets of all $\Gamma \in \mathcal{F}_h$ such that $\Gamma \subset \partial\Omega_w$, $\Gamma \subset \partial\Omega_i$ and $\Gamma \subset \partial\Omega_o$, respectively. Furthermore, we denote by $\mathcal{F}_h^D$ the set of all $\Gamma \in \mathcal{F}_h$ where the Dirichlet type of boundary conditions is prescribed at least for one component of $\boldsymbol{w}$ (i.e., $\mathcal{F}_h^D = \mathcal{F}_h^w \cup \mathcal{F}_h^i$) and by $\mathcal{F}_h^N$ the set of all $\Gamma \in \mathcal{F}_h$ where only the Neumann boundary conditions are prescribed (i.e., $\mathcal{F}_h^N = \mathcal{F}_h^o$). For a shorter notation we put $\mathcal{F}_h^{io} = \mathcal{F}_h^i \cup \mathcal{F}_h^o$, $\mathcal{F}_h^{ID} = \mathcal{F}_h^I \cup \mathcal{F}_h^D$ and $\mathcal{F}_h^{DN} = \mathcal{F}_h^D \cup \mathcal{F}_h^N = \mathcal{F}_h^w \cup \mathcal{F}_h^i \cup \mathcal{F}_h^o$.

Finally, for each $\Gamma \in \mathcal{F}_h$ we define a unit normal vector $\boldsymbol{n}_\Gamma$. We assume that for $\Gamma \in \mathcal{F}_h^{DN}$ the vector $\boldsymbol{n}_\Gamma$ has the same orientation as the outer normal of $\partial\Omega$. For $\boldsymbol{n}_\Gamma$, $\Gamma \in \mathcal{F}_h^I$, the orientation is arbitrary but fixed for each face.

3.2 Discontinuous finite element spaces

To each $K \in \mathcal{T}_h$, we assign a positive integer p_K (local polynomial degree). Then we define the vector $\mathsf{p} = \{p_K, K \in \mathcal{T}_h\}$. Over the triangulation $\mathcal{T}_h$ we define the finite dimensional space of discontinuous piecewise polynomial functions associated with the vector p by

$$S_{h\mathsf{p}} = \{v;\ v \in L^2(\Omega),\ v|_K \in P_{p_K}(K)\ \forall K \in \mathcal{T}_h\}, \tag{3}$$

where $P_{p_K}(K)$ denotes the space of all polynomials on K of degree $\leq p_K$, $K \in \mathcal{T}_h$. We seek the approximate solution in the space $\boldsymbol{S}_{h\mathsf{p}} = S_{h\mathsf{p}} \times \ldots \times S_{h\mathsf{p}}$ $\quad (d+2$ times).

For each $\Gamma \in \mathcal{F}_h^I$ there exist two elements $K_p, K_n \in \mathcal{T}_h$ such that $\Gamma \subset K_p \cap K_n$. We use a convention that K_n lies in the direction of $\boldsymbol{n}_\Gamma$ and K_p in the opposite direction of $\boldsymbol{n}_\Gamma$. Then for $v \in S_{h\mathsf{p}}$, we introduce the notation: $v|_\Gamma^{(p)}$ is the trace of $v|_{K_p}$ on Γ, $v|_\Gamma^{(n)}$ is the trace of $v|_{K_n}$ on Γ, and $\langle v \rangle_\Gamma := \frac{1}{2}\left(v|_\Gamma^{(p)} + v|_\Gamma^{(n)}\right)$, $[v]_\Gamma := v|_\Gamma^{(p)} - v|_\Gamma^{(n)}$.

For $\Gamma \in \mathcal{F}_h^{DN}$ there exists element $K_p \in \mathcal{T}_h$ such that $\Gamma \subset K_p \cap \partial\Omega$. Then for $v \in \boldsymbol{S}_{h\mathsf{p}}$, we denote by $v|_\Gamma^{(p)}$ the trace of $v|_{K_p}$ on Γ and $\langle v \rangle_\Gamma = [v]_\Gamma = v|_\Gamma^{(p)}$. By $v|_\Gamma^{(n)}$, $\Gamma \in \mathcal{F}_h^D \cup \mathcal{F}_h^N$, we formally denote the trace of v on Γ from the exterior of Ω given either by a boundary condition or by an extrapolation from the interior of Ω.

In case that $[\cdot]_\Gamma$ and $\langle \cdot \rangle_\Gamma$ are arguments of $\int_\Gamma \ldots \, \mathrm{d}S$, $\Gamma \in \mathcal{F}_h$ we omit the subscript Γ and write simply $[\cdot]$ and $\langle \cdot \rangle$, respectively.

3.3 Discretization of the Navier-Stokes equations

In this section, we recall the *backward difference formula – discontinuous Galerkin finite element* (BDF-DGFE) method for the solution of the Navier-Stokes equations (1) presented in [9]. For $\boldsymbol{w}_h, \bar{\boldsymbol{w}}_h, \boldsymbol{\varphi}_h \in \boldsymbol{S}_{h\mathsf{p}}$, we define the forms, which represent the *interior penalty Galerkin* (IPG) discretization of (1), namely

$$
\begin{aligned}
\boldsymbol{c}_h(\bar{\boldsymbol{w}}_h, \boldsymbol{w}_h, \boldsymbol{\varphi}_h) := &- \sum_{K\in\mathcal{T}_h} \int_K \sum_{s=1}^{d} \boldsymbol{A}_s(\bar{\boldsymbol{w}}_h)\boldsymbol{w}_h \cdot \frac{\partial \boldsymbol{\varphi}_h}{\partial x_s}\,\mathrm{d}x \\
&+ \sum_{\Gamma\in\mathcal{F}_h^I} \int_\Gamma \left(\boldsymbol{P}^+\left(\langle \bar{\boldsymbol{w}}_h\rangle, \boldsymbol{n}\right) \boldsymbol{w}_h|_\Gamma^{(p)} + \boldsymbol{P}^-\left(\langle \bar{\boldsymbol{w}}_h\rangle, \boldsymbol{n}\right) \boldsymbol{w}_h|_\Gamma^{(n)} \right) \cdot [\boldsymbol{\varphi}_h]\mathrm{d}S \\
&+ \sum_{\Gamma\in\mathcal{F}_h^{io}} \int_\Gamma \left(\boldsymbol{P}^+\left(\langle \bar{\boldsymbol{w}}_h\rangle, \boldsymbol{n}\right) \boldsymbol{w}_h|_\Gamma^{(p)} \right) \cdot [\boldsymbol{\varphi}_h]\,\mathrm{d}S + \sum_{\Gamma\in\mathcal{F}_h^{w}} \int_\Gamma \boldsymbol{F}_W(\bar{\boldsymbol{w}}_h, \boldsymbol{w}_h, \boldsymbol{n}) \cdot \boldsymbol{\varphi}_h\,\mathrm{d}S, \\
&+ \sum_{K\in\mathcal{T}_h} \int_K \sum_{s,k=1}^{d} \left(\boldsymbol{K}_{s,k}(\bar{\boldsymbol{w}}_h) \frac{\partial \boldsymbol{w}_h}{\partial x_k} \right) \cdot \frac{\partial \boldsymbol{\varphi}_h}{\partial x_s}\,\mathrm{d}x \\
&- \sum_{\Gamma\in\mathcal{F}_h^{ID}} \int_\Gamma \sum_{s=1}^{d} \left\langle \sum_{k=1}^{d} \boldsymbol{K}_{s,k}(\bar{\boldsymbol{w}}_h) \frac{\partial \boldsymbol{w}_h}{\partial x_k} \right\rangle n_s \cdot [\boldsymbol{\varphi}_h]\,\mathrm{d}S \\
&-\theta \sum_{\Gamma\in\mathcal{F}_h^{ID}} \int_\Gamma \sum_{s=1}^{d} \left\langle \sum_{k=1}^{d} \boldsymbol{K}_{s,k}^{\mathrm{T}}(\bar{\boldsymbol{w}}_h) \frac{\partial \boldsymbol{\varphi}_h}{\partial x_k} \right\rangle n_s \cdot [\boldsymbol{w}_h]\,\mathrm{d}S + \sum_{\Gamma\in\mathcal{F}_h^{ID}} \int_\Gamma \sigma [\boldsymbol{w}_h] \cdot [\boldsymbol{\varphi}_h]\,\mathrm{d}S
\end{aligned}
\tag{4}
$$

and

$$
\begin{aligned}
\tilde{\boldsymbol{b}}_h(\bar{\boldsymbol{w}}_h, \boldsymbol{\varphi}_h) := &- \sum_{\Gamma\in\mathcal{F}_h^{io}} \int_\Gamma \left(\boldsymbol{P}^-\left(\langle \bar{\boldsymbol{w}}_h\rangle, \boldsymbol{n}\right) \bar{\boldsymbol{w}}_h|_\Gamma^{(n)} \right) \cdot [\boldsymbol{\varphi}_h]\,\mathrm{d}S, \\
&-\theta \sum_{\Gamma\in\mathcal{F}_h^{D}} \int_\Gamma \sum_{s,k=1}^{d} \boldsymbol{K}_{s,k}^{\mathrm{T}}(\bar{\boldsymbol{w}}_h) \frac{\partial \boldsymbol{\varphi}_h}{\partial x_k}\, n_s \cdot \boldsymbol{w}_B\,\mathrm{d}S + \sum_{\Gamma\in\mathcal{F}_h^{D}} \int_\Gamma \sigma \boldsymbol{w}_B \cdot \boldsymbol{\varphi}_h\,\mathrm{d}S,
\end{aligned}
\tag{5}
$$

where $\boldsymbol{A}_s(\cdot)$ are the Jacobi matrices of the mappings $\boldsymbol{f}_s,\ s = 1, \ldots, d$, $\boldsymbol{P}^\pm$ are the positive and negative parts of the matrix $\boldsymbol{P}(\boldsymbol{w}, \boldsymbol{n}) := \sum_{s=1}^{d} \boldsymbol{A}_s(\boldsymbol{w}) n_s$, which define the Vijayasundaram numerical flux [23] used for the approximation of inviscid fluxes trough $\Gamma \in \mathcal{F}_h$. Moreover,

$$
\boldsymbol{F}_W(\bar{\boldsymbol{w}}_h, \boldsymbol{w}_h, \boldsymbol{n}) = (\gamma - 1) D\boldsymbol{F}_W(\bar{\boldsymbol{w}}_h, \boldsymbol{n}) \boldsymbol{w}_h, \tag{6}
$$

where $D\boldsymbol{F}_W(\boldsymbol{w}, \boldsymbol{n})$ is a $(d+2) \times (d+2)$ Jacobi matrix of $(0, \mathrm{p}n_1, \ldots, \mathrm{p}n_d, 0)^{\mathrm{T}}$, where p is pressure, see [8], [9] or [11]. Furthermore, $\bar{\boldsymbol{w}}|_\Gamma^{(n)} = LRP(\bar{\boldsymbol{w}}|_\Gamma^{(p)}, \boldsymbol{w}_D, \boldsymbol{n}_\Gamma),\ \Gamma \in \mathcal{F}_h^{io}$ where $LRP(\cdot, \cdot, \cdot)$ represents a solution of the *local Riemann problem* considered on edge $\Gamma \in \mathcal{F}_h^{io}$ and $\boldsymbol{w}_D$ is a given state vector (e.g. from far-field boundary conditions), see [10]. The

state vector $\boldsymbol{w}_B$ prescribed on $\partial\Omega_i \cup \partial\Omega_w$ is given by the boundary conditions, see [8] or [9].

The value of θ appearing in (4) – (5) is equal to 1, -1 or 0 depending on the variant of interior penalty method, namely $\theta = 1$ for *symmetric interior penalty Galerkin* (SIPG), $\theta = -1$ for *non-symmetric interior penalty Galerkin* (NIPG), $\theta = 0$ for *incomplete interior penalty Galerkin* (IIPG). Finally, the penalty parameter σ is chosen by $\sigma|_\Gamma = C_W/(\mathrm{diam}(\Gamma)\,\mathrm{Re}), \quad \Gamma \in \mathcal{F}_h^{ID}$, where Re is the Reynolds number of the flow and $C_W > 0$ is a suitable constant whose choice depends on the used variant of the DGFE method (NIPG, IIPG or SIPG) and the degree of polynomial approximation, see [9].

The forms $\boldsymbol{c}_h$ and $\tilde{\boldsymbol{c}}_h$ make sense not only for piecewise polynomial functions but also for functions from $H^2(\Omega, \mathcal{T}_h) := \{\boldsymbol{\varphi}; \boldsymbol{\varphi}|_K \in (H^2(K))^{d+2}\ \forall\, K \in \mathcal{T}_h\}$ where $H^2(K)$ is the standard Sobolev space over K. It is possible to show (see, e.g., [8], [9]) that if $\boldsymbol{w} : \Omega \times (0,T) \to I\!R^{d+2}$ is a sufficiently regular function satisfying the Navier-Stokes equations (1) and the corresponding initial and boundary conditions then

$$\frac{\mathrm{d}}{\mathrm{d}t}\,(\boldsymbol{w}, \boldsymbol{\varphi}) + \boldsymbol{c}_h\,(\boldsymbol{w}, \boldsymbol{w}, \boldsymbol{\varphi}) = \tilde{\boldsymbol{c}}_h\,(\boldsymbol{w}, \boldsymbol{\varphi}) \quad \forall \boldsymbol{\varphi} \in \boldsymbol{S}_{h\mathsf{p}}, \tag{7}$$

where $(\cdot,\cdot)$ denotes L^2-scalar product over Ω.

Similarly, if $\boldsymbol{w} : \Omega \to I\!R^{d+2}$ is a sufficiently regular function satisfying the stationary Navier-Stokes equations and the corresponding boundary conditions then

$$\boldsymbol{c}_h\,(\boldsymbol{w}, \boldsymbol{w}, \boldsymbol{\varphi}) = \tilde{\boldsymbol{c}}_h\,(\boldsymbol{w}, \boldsymbol{\varphi}) \quad \forall \boldsymbol{\varphi} \in \boldsymbol{S}_{h\mathsf{p}}. \tag{8}$$

Now, we introduce the space semi-discretization of (CFP). Let $C^1([0,T]; \boldsymbol{S}_{h\mathsf{p}})$ denote the space of continuously differentiable mappings of the interval $[0,T]$ into $\boldsymbol{S}_{h\mathsf{p}}$.

Definition 1 *Function $\boldsymbol{w}_h \in C^1([0,T]; \boldsymbol{S}_{h\mathsf{p}})$ is called the* semi-discrete solution *of (CFP), if*

$$\begin{aligned} &\text{a)} \quad \left(\frac{\partial \boldsymbol{w}_h(t)}{\partial t}, \boldsymbol{\varphi}_h\right) + \boldsymbol{c}_h(\boldsymbol{w}_h(t), \boldsymbol{w}_h(t), \boldsymbol{\varphi}_h) = \tilde{\boldsymbol{c}}_h(\boldsymbol{w}_h(t), \boldsymbol{\varphi}_h) \quad \forall\, \boldsymbol{\varphi}_h \in \boldsymbol{S}_{h\mathsf{p}}\ \forall\, t \in (0,T), \\ &\text{b)} \quad \boldsymbol{w}_h(0) = \boldsymbol{w}_h^0, \end{aligned} \tag{9}$$

where $\boldsymbol{w}_h^0 \in \boldsymbol{S}_{h\mathsf{p}}$ denotes an $\boldsymbol{S}_{h\mathsf{p}}$-approximation of the initial condition.

The problem (9), a) – b) represents a system of ordinary differential equations (ODEs) for $\boldsymbol{w}_h(t)$ which has to be discretized in time by a suitable method. Since these ODEs represent a stiff system, a use of a (semi-)implicit method is advantageous. Therefore, we employ the *semi-implicit* technique developed in [9] and [11] which is based on the linearity of the form $\boldsymbol{c}_h(\cdot,\cdot,\cdot)$ with respect to its second argument following from expressions (4) – (5). Hence, for the first order scheme with respect to time, we approximate the time derivative term in (9), a) by backward Euler method, the second argument of $\boldsymbol{c}_h(\cdot,\cdot,\cdot)$ is treated implicitly and the first one explicitly.

Let $0 = t_0 < t_1 < t_2 < \dots t_r = T$ be a partition of the time interval $(0, T)$, $\tau_k := t_k - t_{k-1}$, and $\boldsymbol{w}_h^k \in \boldsymbol{S}_{h\mathsf{p}}$ denotes a piecewise polynomial approximation of $\boldsymbol{w}_h(t_k)$, $k = 0, 1, \dots, r$. We define the following scheme.

Definition 2 *The* approximate solution *of (*CFP*) by the semi-implicit DGFE scheme is defined as functions* $\boldsymbol{w}_{h,k}$, $k = 1, \dots, r$, *satisfying the conditions*

$$\begin{aligned}
&a) \quad \boldsymbol{w}_{h,k} \in \boldsymbol{S}_{h\mathsf{p}}, && (10)\\
&b) \quad \frac{1}{\tau_k}\left(\boldsymbol{w}_{h,k} - \boldsymbol{w}_{h,k-1},\ \boldsymbol{\varphi}_h\right) + \boldsymbol{c}_h\left(\boldsymbol{w}_{h,k-1},\ \boldsymbol{w}_{h,k},\ \boldsymbol{\varphi}_h\right) = \tilde{\boldsymbol{c}}_h\left(\boldsymbol{w}_{h,k-1},\ \boldsymbol{\varphi}_h\right) \quad \forall\, \boldsymbol{\varphi}_h \in \boldsymbol{S}_{h\mathsf{p}},\\
&c) \quad \boldsymbol{w}_{h,0} \in S_{h\mathsf{p}} \text{ is an approximation of } \boldsymbol{w}^0.
\end{aligned}$$

The method (10) is a first order scheme with respect to the time which is sufficient for steady-state problems. Otherwise, it is possible to use multi-step backward difference formulae for the time discretization, see [9]. Then all considerations presented in this paper have to be slightly modified.

The problem (10), a) – c) represents a linear algebraic system for each $k = 1, \dots, r$, whose solution is discussed in Section 4. Numerical experiments show that the resulting semi-implicit DGFE method is practically unconditionally stable, i.e., the size of the time step can be chosen very large, see [9].

Finally, we introduce the discrete problem for the stationary Navier-Stokes equations.

Definition 3 *Function* $\boldsymbol{w}_h \in \boldsymbol{S}_{h\mathsf{p}}$ *is called the* discrete solution *of (*sCFP*), if*

$$\boldsymbol{c}_h(\boldsymbol{w}_h, \boldsymbol{w}_h, \boldsymbol{\varphi}_h) = \tilde{\boldsymbol{c}}_h(\boldsymbol{w}_h, \boldsymbol{\varphi}_h) \qquad \forall\, \boldsymbol{\varphi}_h \in \boldsymbol{S}_{h\mathsf{p}}. \tag{11}$$

We already mentioned in Introduction, the non-stationary formulation (9) will be used for the solution of (sCFP).

4 Solution strategy

In this section, we present an efficient solution strategy of the (stationary) discrete problem (11). However, its direct solution causes some troubles mentioned bellow. Hence we employ the non-stationary formulation (10) for its solution.

4.1 Linear algebra representation

4.1.1 Basis of $\boldsymbol{S}_{h\mathsf{p}}$

Let us introduce an index set $I \subset \boldsymbol{Z}^+$(=set of all positive integers) numbering elements $K \in \mathcal{T}_h$, i.e., $\mathcal{T}_h = \{K_\mu,\ \mu \in I\}$. By $p_\mu = p_{K_\mu}$ we denote the degree of polynomial approximation on element K_μ, $\mu \in I$. Since $\boldsymbol{S}_{h\mathsf{p}}$ is a space of discontinuous piecewise

polynomial functions, it is possible to consider a set of linearly independent polynomial functions on K_μ for each $K_\mu \in \mathcal{T}_h$. Then we define the basis of $\boldsymbol{S}_{h\mathsf{p}}$ by

$$B = \Big\{\boldsymbol{\psi}_{\mu,j};\ \boldsymbol{\psi}_{\mu,j} \in \boldsymbol{S}_{h\mathsf{p}},\ \mathrm{supp}(\boldsymbol{\psi}_{\mu,j}) \subseteq K_\mu, \qquad (12)$$
$$\boldsymbol{\psi}_{\mu,j} \text{ are linearly independent for } j = 1,\dots,\mathsf{dof}_\mu,\ \mu \in I\Big\},$$

where $\mathsf{dof}_\mu = (d+2)/d!\,\Pi_{j=1}^{d}(p_\mu + j)$, $\mu \in I$ denotes the number of *local degrees of freedom* for each element $K_\mu \in \mathcal{T}_h$. By dof, we denote the dimension of $\boldsymbol{S}_{h\mathsf{p}}$ (=number of elements of the basis B) which is equal to $\mathsf{dof} = \sum_{\mu\in I} \mathsf{dof}_\mu$.

Therefore, a function $\boldsymbol{w}_{h,k} \in \boldsymbol{S}_{h\mathsf{p}}$ can be written in the form

$$\boldsymbol{w}_{h,k}(x) = \sum_{\mu\in I}\sum_{j=1}^{\mathsf{dof}_\mu} \xi_{k,\mu,j}\boldsymbol{\psi}_{\mu,j}(x), \quad x \in \Omega,\ k = 0,1,\dots,r, \qquad (13)$$

where $\xi_{k,\mu,j} \in I\!R,\ j = 1,\dots,\mathsf{dof}_\mu,\ \mu \in I,\ k = 0,\dots,r$. Moreover, for $\boldsymbol{w}_{h,k} \in \boldsymbol{S}_{h\mathsf{p}}$, we define a vector of its basis coefficients by $\boldsymbol{\xi}_k = \{\xi_{k,\mu,j}\}_{j=1,\dots,\mathsf{dof}_\mu}^{\mu\in I} \in I\!R^{\mathsf{dof}},\ k = 0,1,\dots,r$. Therefore, using (13), we have an isomorphism

$$\boldsymbol{w}_{h,k} \in \boldsymbol{S}_{h\mathsf{p}} \quad \longleftrightarrow \quad \boldsymbol{\xi}_k \in I\!R^{\mathsf{dof}}. \qquad (14)$$

4.1.2 Linear algebraic systems

Using isomorphism (14), problem (10) can be written in the matrix form

$$\underbrace{\left(\frac{1}{\tau_k}\boldsymbol{M} + \boldsymbol{C}(\boldsymbol{\xi}_{k-1})\right)}_{=:\boldsymbol{A}_k}\boldsymbol{\xi}_k = \underbrace{\frac{1}{\tau_k}\boldsymbol{m}(\boldsymbol{\xi}_{k-1}) + \boldsymbol{q}(\boldsymbol{\xi}_{k-1})}_{=:\boldsymbol{d}_k}, \quad k = 1,\dots,r, \qquad (15)$$

where the matrix $\boldsymbol{M}$ is a block-diagonal *mass matrix* given by

$$\boldsymbol{M} = \{(\boldsymbol{\psi}_{\mu,i}, \boldsymbol{\psi}_{\mu,j})\}_{\mu,\nu\in I}^{i=1,\dots,\mathsf{dof}_\mu, j=1,\dots,\mathsf{dof}_\nu},\ \mu,\nu \in I, \qquad (16)$$

the matrix $\boldsymbol{C}$ is the *"flux" matrix* corresponding to form $\boldsymbol{c}_h$ defined by

$$\boldsymbol{C}(\boldsymbol{\xi}_{k-1})) = \{\boldsymbol{c}_h\left(\boldsymbol{w}_{h,k-1},\ \boldsymbol{\psi}_{\nu,j},\ \boldsymbol{\psi}_{\mu,i}\right)\}_{\mu,\nu\in I}^{i=1,\dots,\mathsf{dof}_\mu, j=1,\dots,\mathsf{dof}_\nu}, \qquad (17)$$

$\boldsymbol{m} \in I\!R^{\mathsf{dof}}$ represents the "explicit" part of the approximation of the time derivative in (10), b) defined by

$$\boldsymbol{m}(\boldsymbol{\xi}_{k-1}) = \boldsymbol{M}\boldsymbol{\xi}_{k-1} = \{\left(\boldsymbol{w}_{h,k-1},\ \boldsymbol{\psi}_{\mu,i}\right)\}_{\mu\in I}^{i=1,\dots,\mathsf{dof}_\mu}, \qquad (18)$$

and $\boldsymbol{q} \in I\!R^{\mathsf{dof}}$ represents form $\tilde{\boldsymbol{c}}_h$ in (10), b) given by

$$\boldsymbol{q}(\boldsymbol{\xi}_{k-1}) = \{\tilde{\boldsymbol{c}}_h\left(\boldsymbol{w}_{h,k-1},\ \boldsymbol{\psi}_{\mu,i}\right)\}_{\mu\in I}^{i=1,\dots,\mathsf{dof}_\mu}. \qquad (19)$$

In virtue of the local character of basis B matrix $\boldsymbol{C}$ has a block structure. The matrix element $C_{(\mu,i),(\nu,j)}$ is non-vanishing if $\mu = \nu$ or if elements K_μ and K_ν share an face.

4.2 Abstract solution strategy

In virtue of (15), the stationary discrete problem (11) reads: find $\boldsymbol{\xi} \in I\!R^{\mathsf{dof}}$ such that

$$\boldsymbol{C}(\boldsymbol{\xi})\boldsymbol{\xi} = \boldsymbol{q}(\boldsymbol{\xi}). \tag{20}$$

The problem (20) represents a system of nonlinear algebraic equations. It is possible to define an iterative method for solving (20) by: find $\boldsymbol{\xi}_k \in I\!R^{\mathsf{dof}},\ k = 1, 2, \ldots$ such that

$$\boldsymbol{C}(\boldsymbol{\xi}_{k-1})\boldsymbol{\xi}_k = \boldsymbol{q}(\boldsymbol{\xi}_{k-1}), \qquad k = 1, 2, \ldots, \tag{21}$$

and put $\boldsymbol{\xi} := \lim_{k\to\infty} \boldsymbol{\xi}_k$. However, this simple method works only if the initial guess $\boldsymbol{\xi}_0$ is very close to solution $\boldsymbol{\xi}$ of (20). Otherwise, the iterative process (21) fails since nonphysical solutions appear.

One possibility how to avoid this principle obstacle is a use of the unsteady formulation (15). The idea is natural. We start with a small time step for small k when the approximation $\boldsymbol{\xi}_k$ is far from $\boldsymbol{\xi}$. Then, when $\boldsymbol{\xi}_k$ is approaching to the limit vector $\boldsymbol{\xi}$ for increasing k, we successively increase the size of the time step τ_k. Consequently, the problems (15) lead to (21) for $\tau_k \to \infty$. On the other hand, the non-stationary discretization (15) can be considered as a "relaxation" of method (21) and the ratio $1/\tau_k$ as the relaxation parameter.

It is suitable to employ an iterative solver for the solution of the linear algebraic systems (15) since the solution $\boldsymbol{\xi}_k,\ k = 1, 2, \ldots$ obtained in k^{th}-iteration can be used as initial guess for $\boldsymbol{\xi}_{k+1}$. We employ the GMRES method [22] with the block ILU(0) preconditioner which represents a widely used technique for the solution of non-symmetric sparse linear algebraic systems. The based ILU(0) preconditioner uses the incomplete LU factorization by performing block Gaussian elimination but ignoring blocks which would result in any additional fill of the matrix. Therefore, no additional memory is required.

Now, we are ready to introduce

Abstract algorithm (AA)

1. let $\boldsymbol{\xi}_0 \longleftrightarrow \boldsymbol{w}_h^0$ be given
2. for $k = 1$ to r
   (a) set τ_k
   (b) from $\boldsymbol{\xi}_{k-1}$ evaluate $\boldsymbol{A}_k(\boldsymbol{\xi}_{k-1})$ and $\boldsymbol{d}_k(\boldsymbol{\xi}_{k-1})$
   (c) solve $\boldsymbol{A}_k(\boldsymbol{\xi}_{k-1})\boldsymbol{\xi}_k = \boldsymbol{d}_k(\boldsymbol{\xi}_{k-1})$ by:
      i. $\boldsymbol{\xi}_k^0 := \boldsymbol{\xi}_{k-1}$
      ii. $\boldsymbol{\xi}_k^{l+1} := \mathsf{GMRES\_iter}(\boldsymbol{\xi}_k^l),\ \ l = 1, \ldots, s_k$
      iii. $\boldsymbol{\xi}_k := \boldsymbol{\xi}_k^{s_k}$
3. $\boldsymbol{\xi} := \boldsymbol{\xi}_r$.

Here, GMRES_iter$(\cdot)$ formally denotes a performance of one GMRES step with the block ILU(0) preconditioner.

In the abstract algorithm (AA), there are still open questions how to choose the total number of time levels r, the size of $\tau_k,\ k = 1, \dots, r$ and the number of GMRES steps s_k at each time level. Therefore, in order to define a real algorithm we have to specify the *algorithm settings*, namely

- *stopping steady-state criterion*, i.e., when to stop the global iterative loop,
- *GMRES stopping criterion*, i.e., how many GMRES steps have to be employed at each time level,
- *choice of the time step* τ_k.

Our goal is to define the previous algorithm settings in order to achieve the final limit vector $\boldsymbol{\xi}$ as soon as possible (measured in terms of the computational time). In the following, we discuss these aspects separately.

4.3 Algorithm settings

4.3.1 Stopping steady-state criterion

In virtue of (20), it is possible to employ the *steady-state residual criterion*

$$\|\boldsymbol{C}(\boldsymbol{\xi}_k)\boldsymbol{\xi}_k - \boldsymbol{q}(\boldsymbol{\xi}_k)\|_{\ell^2} \leq \mathrm{TOL}, \tag{22}$$

which is independent of τ_k and measures the residuum of the nonlinear algebraic system (20). However, the residuum (22) depends on the size of the computational domain Ω, on the magnitude of components of $\boldsymbol{w}_h$, etc. Therefore, from the practical reasons, we use the *relative residuum steady-state criterion*

$$\mathrm{SSres}(k) := \frac{\|\boldsymbol{C}(\boldsymbol{\xi}_k)\boldsymbol{\xi}_k - \boldsymbol{q}(\boldsymbol{\xi}_k)\|_{\ell^2}}{\|\boldsymbol{C}(\boldsymbol{\xi}_0)\boldsymbol{\xi}_0 - \boldsymbol{q}(\boldsymbol{\xi}_0)\|_{\ell^2}} \leq \mathrm{TOL}, \tag{23}$$

which already does not suffer from the mentioned drawbacks.

Moreover, we employ the stopping criteria which follow from the physical background of the considered problem. Many often, we are interested in the *aerodynamic coefficients* of the considered flow, namely coefficients of *drag* (c_D), *lift* (c_L) and *momentum* (c_M). Then the natural choice is to stop global iterative loops when these coefficients achieve a given tolerance, e.g.,

$$\Delta c_x(k) \leq \mathrm{tol}, \qquad \Delta c_x(k) := \max_{l=0.9k,\dots,k} c_x(l) - \min_{l=0.9k,\dots,k} c_x(l), \tag{24}$$

where tol is a given tolerance, subscript x takes the values D, L and M (drag, lift, momentum), $c_x(k)$ is the value of the corresponding aerodynamic coefficient at k^{th}-time level and the minimum and maximum in (24) are taken over the last 10% of the number of time levels.

4.3.2 GMRES stopping criterion

In this section we deal with the stopping criterion of the inner loops in (AA), i.e., when to stop the GMRES iterative process at each time level $k = 1, \dots, r$. Since our aim is to obtain the final steady-state solution $\boldsymbol{\xi}$ from (AA) as fast as possible it does not make sense to solve the linear algebraic systems (15) too precisely.

In [5], the so-called *inexact Newton method* was proposed for the solution of a system of nonlinear algebraic equations. The main idea is that the linear algebraic systems (arising from the Newton method) are solved by an iterative solver till the residuum is only a few times smaller than the residuum of the initial guess of the solution (taken from the previous level). Using this idea, we use the following stopping criterion for the GMRES method at each time level:

$$\|\boldsymbol{A}_k(\boldsymbol{\xi}_{k-1})\boldsymbol{\xi}_k - \boldsymbol{d}_k(\boldsymbol{\xi}_{k-1})\| \le \delta_k \|\boldsymbol{A}_k(\boldsymbol{\xi}_{k-1})\boldsymbol{\xi}_{k-1} - \boldsymbol{d}_k(\boldsymbol{\xi}_{k-1})\|, \quad k = 1, \dots, r, \tag{25}$$

where $\delta_k \in (0, 1)$ is a given value, the left-hand side is the residuum and the term on the right-hand side can be considered either as the *consistency residuum* from the previous time level or the *initial residuum* since the solution of the previous time level is taken as an initial solution on the next time level. Therefore, the iterative process is stopped if the residuum is $1/\delta_k$-times smaller than the initial one.

However, in order to save the computational time, we employ the stopping criterion in the form

$$\|\hat{\boldsymbol{P}}(\boldsymbol{A}_k(\boldsymbol{\xi}_{k-1})\boldsymbol{\xi}_k - \boldsymbol{d}_k(\boldsymbol{\xi}_{k-1}))\| \le \delta_k \|\hat{\boldsymbol{P}}(\boldsymbol{A}_k(\boldsymbol{\xi}_{k-1})\boldsymbol{\xi}_{k-1} - \boldsymbol{d}_k(\boldsymbol{\xi}_{k-1}))\|, \quad k = 1, \dots, r, \tag{26}$$

since the norm on the left-hand side of (26) is available at each GMRES step and thus we need not to evaluate the residuum at the left-hand-side of (25). Numerical experiments show that both stopping criteria (25) and (26) have very similar behavior.

In [13], there were presented two choices of δ_k and the corresponding orders of convergence of the inexact Newton method were proved. However, our numerical experiments show that the efficiency of the method only weakly depends on value δ_k chosen around 1/2. Therefore, we put $\delta_k = \delta = 1/2,\ k = 1, \dots, r$ in our algorithm.

4.3.3 Choice of the time step

The strategy of the choice of the time step has a great influence to the efficiency of the discussed method. We already mentioned that the semi-implicit time discretization allows to choose the time step many times larger than an explicit scheme. On the other hand, at the beginning of the computation, we usually start from an unphysical initial condition and then too large time step can cause a fail of the computational process. Therefore, our aim is to construct sufficiently robust algorithm which automatically increase the time step from small values at the beginning of the computation to the larger ones.

We use a strategy which is based on an very low cost estimation of the local discretization error. Let us consider ODE $y' = f(y)$, where $y : [0,T] \to I\!R$ and $f : I\!R \to I\!R$. We denote by $y_k \approx y(t_k)$ an approximation of the solution y at $t_k,\ k = 1,\dots,r$. The local discretization error L_k of the *backward Euler method* is given by

$$L_k := y(t_k) - y_k \approx \frac{1}{2}\tau_k^2 y''(\theta_k), \qquad \theta_k \in (t_{k-1}, t_k), \tag{27}$$

where y'' denotes the second order derivative of y. Now, we define a quadratic function $\tilde{y} : (t_{k-2}, t_k) \to I\!R$ such that $\tilde{y}(t_{k-l}) = y_{k-l}, \quad l = 0,1,2$. The second order derivative of $\tilde{y}$ is constant on (t_{k-2}, t_k) and hence we put

$$L_k \approx L_k^{\mathrm{app}} := \frac{1}{2}\tau_k^2 \tilde{y}''. \tag{28}$$

Let $\omega > 0$ be a given tolerance for the local discretization error. Our aim is to choose the time step as large as possible but $L_k \le \omega,\ k = 1,\dots,r$. Using (28), we obtain a relation for the optimal size of τ_k by

$$\tau_k^{\mathrm{opt}} := \tau_k \left(\frac{\omega}{L_k^{\mathrm{app}}} \right)^{1/2}. \tag{29}$$

Hence, if $L_k \le \omega$ we put $\tau_{k+1} := \tau_k^{\mathrm{opt}}$ and proceed to the next time level. Otherwise, we repeat k^{th}-time step with $\tau_k := \tau_k^{\mathrm{opt}}$. From practical reasons, we restrict the increase of the time step by the condition $\tau_{k+1} \le 2.5\tau_k$. Finally, the first two time steps are chosen in this way that τ_0 and τ_1 correspond to the time steps used for the explicit time discretization with CFL = 0.5, see [14]. This approach can be simply extend to a system of ODEs component-wise.

4.4 New solution strategy

We summarize the solution strategy for the solution of (10). We use the algorithm (AA) where

i) the global loop is stopped if the steady state conditions (23) and (24) are satisfied,

ii) the size of the time step is chosen by the algorithm presented in Section 4.3.3,

iii) the GMRES iterations are performed till the stopping criterion (26) is satisfied.

5 Numerical experiments

In this section, we present a set of numerical experiments in order to demonstrate the efficiency of the method, namely the relative computational times necessary for the setting and the solution of the linear algebraic systems. Moreover, we present two additional numerical examples demonstrating the robustness of the solution strategy with respect to the flow regime.

P_k	$\#\mathcal{T}_h$	dof	preparing $\boldsymbol{A}_k, \mathrm{d}_k$	solving $\boldsymbol{A}_k\boldsymbol{\xi}_k = \mathrm{d}_k$	CPU(s)	$\frac{\text{CPU(s)}}{\text{dof}}$	Σ
P_1	1 666	19 992	56%	43%	55.8	2.79E-03	4
P_1	2 394	28 728	57%	42%	81.1	2.82E-03	4
P_1	3 530	42 360	61%	38%	188.7	4.45E-03	6
P_1	4 214	50 568	65%	34%	224.0	4.43E-03	7
P_2	1 666	39 984	56%	43%	139.4	3.49E-03	3
P_2	2 394	57 456	58%	41%	243.3	4.23E-03	3
P_2	3 530	84 720	56%	43%	424.2	5.01E-03	3
P_2	4 214	101 136	62%	37%	571.3	5.65E-03	5
P_3	1 666	66 640	58%	41%	442.2	6.64E-03	3
P_3	2 394	95 760	56%	43%	602.4	6.29E-03	3
P_3	3 530	141 200	59%	40%	1 274.9	9.03E-03	3
P_3	4 214	168 560	63%	36%	2 158.4	1.28E-02	5

Table 1: Relative computational times for the preparing of the linear algebraic systems and their solution by the new solution strategy, Σ denotes an average number of GMRES steps per each time level

5.1 Efficiency of the algorithm

We consider a laminar viscous subsonic flow around the NACA 0012 profile with inlet Mach number $M_{\text{inlet}} = 0.5$, the angle of attack $\alpha = 2^\circ$ and the Reynolds number $\text{Re} = 5\,000$. We employ the IIPG variant of DGFE method with the penalty parameter $C_W = 200$ and numerical experiments were carried out by P_1, P_2 and P_3 polynomial approximations on four triangular unstructured grids having 1 666, 2 394, 3 530 and 4 214 elements. These meshes were generated by the ANGENER code [7], similarly as the grid from Figure 1. We employ the stopping criteria (23) with $\text{TOL} = 10^{-4}$ and (24) with $\text{tol} = 10^{-4}$.

In order to demonstrate the efficiency of the new solution strategy, we compare the relative computational times necessary for the preparing of the linear algebraic systems (i.e., the evaluations of the matrices $\boldsymbol{A}_k$ and the right-hand side $\boldsymbol{d}_k$, $k = 1, \dots, r$ using (15) – (19)) and themselves solutions with the aid of this strategy. We employ the values $\delta = 0.5$ in (26) and $\omega = 0.5$ in (29). Table 1 shows the relative computational times for the preparing of the linear algebraic systems and their solution by the new solution strategy for all grids and all degrees of polynomial approximations. We observe that the solution of the sequence of the linear algebraic systems (15) requires less than 50% of the computational time. Therefore, our strategy is in fact optimal since any additional increase of the efficiency of the solution of (15) does not cause any essential increase of the efficiency of the global scheme (10). An additional acceleration the scheme (10) would require also an acceleration of the setting of matrices $\boldsymbol{A}_k$, $k = 1, \dots, r$, e.g., by a matrix-free implementation. Last three columns show the total computational times in seconds, the computational time per one degree of freedom (dof) in seconds and the average number of GMRES steps performed at each time level.

Moreover, we compare the proposed strategy with the explicit time discretization from

case	method	P_1 CPU time	P_1 memory	P_3 CPU time	P_3 memory
inviscid	explicit [8]	6 194 s	6 MB	—	41 MB
	semi-implicit [9]	232 s	34 MB	2 283 s	177 MB
	new semi-implicit	47 s	30 MB	226 s	168 MB
viscous	explicit [8]	11 680 s	5 MB	—	38 MB
	semi-implicit [9]	362 s	25 MB	2 292 s	172 MB
	new semi-implicit	97 s	24 MB	613 s	162 MB

Table 2: Comparison of the computational times and the memory requirements of the explicit, semi-implicit and new semi-implicit time discretization

[8] and our former semi-implicit time discretization from [9]. We consider the viscous flow and also the limit inviscid flow ($\mathrm{Re} \to \infty$) with the same data setting. Table 2 shows a comparison of the computational times and the memory requirements of these three approaches for the inviscid and viscous flows using P_1 and P_3 polynomial approximation. We simply observe a significant decrease of the computational times. On the other hand, semi-implicit approaches requires more memory since the matrices are stored in our implementations.

5.2 Additional numerical examples

In order to demonstrate the robustness of the presented approach with respect to the flow regime, we present two additional examples from [21].

5.2.1 Transonic viscous flow

We consider a transonic flow around the NACA 0012 profile with the inlet Mach number $M_{\mathrm{inlet}} = 0.85$, angle of attack $\alpha = 0°$ and the Reynolds number $\mathrm{Re} = 2\,000$. We employ a triangular grid from Figure 1, P_3 polynomial approximation and the algorithm (AA) with $\delta = 0.5$ in (26) and $\omega = 0.5$ in (29).

We observe that the computational time for the solution of the corresponding linear algebraic systems (15) is shorter than the computational time for their preparation. Moreover, approximately 5 GMRES steps is performed at each time level. Therefore, the efficiency of the solution strategy (AA) is high similarly as for the subsonic flow presented in Section 5.1. Figure 2 shows the corresponding isolines of the Mach number and the pressure.

5.2.2 Unsteady viscous flow

Finally, we present a transonic flow around the NACA 0012 profile with the inlet Mach number $M_{\mathrm{inlet}} = 0.85$, angle of attack $\alpha = 0°$ and the Reynolds number $\mathrm{Re} = 10\,000$. This problem is more challenging since the flow is unsteady with a periodic propagation of vortexes behind the profiles, see [21]. As we already mention, our solution

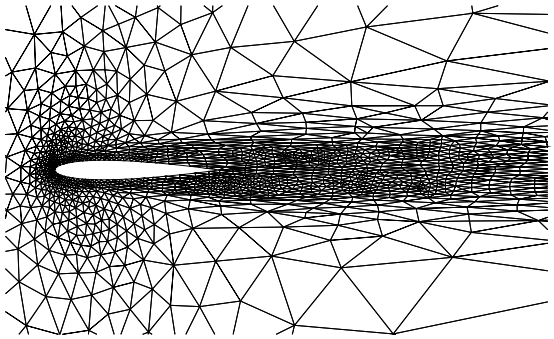

Figure 1: NACA 0012, $M_{\text{inlet}} = 0.85$, $\alpha = 0°$ and $\text{Re} = 2\,000$, triangular grid

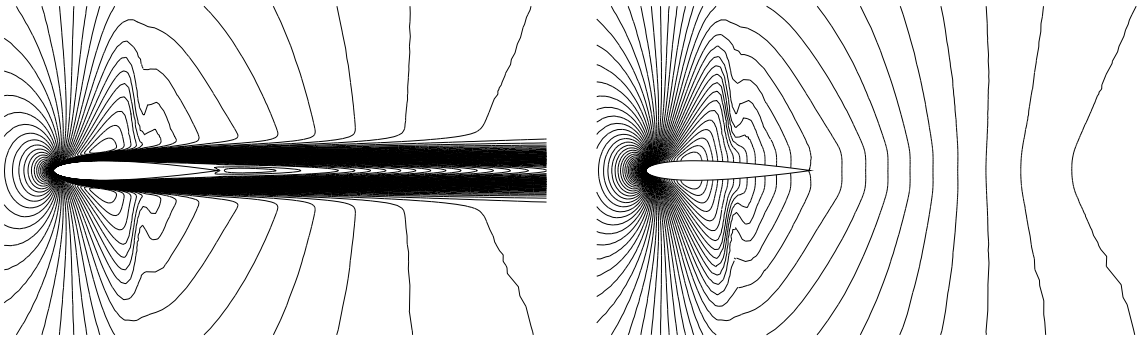

Figure 2: NACA 0012, $M_{\text{inlet}} = 0.85$, $\alpha = 0°$ and $\text{Re} = 2\,000$, isolines of Mach number (top) and isolines of pressure (bottom)

strategy, originally developed for steady flows, can be used without any modification also for unsteady flows. We employ again the value $\delta = 0.5$ in the stopping criterion (26).

On the other hand, the problem is unsteady then it is necessary to choose the time step smaller in order to guarantee an accuracy with respect to the time. Obviously, it is more efficient to use a higher order scheme with respect to time (e.g., BDF formulae as in [9]). This will be a subject of further research. Here, we only demonstrate the capability of our approach to solve unsteady problems. Therefore, we use the adaptive (first order) time step algorithm from Section 4.3.3 and put the tolerance $\omega = 0.01$ in (29).

We employed a grid similar to grid from Figure 1 and P_2 polynomial approximation. We carried out a computation for the physical (dimensionless) time $t \in (0, 90)$. Figure 3 (top) shows the dependence of the lift coefficient c_L on time for $t \in (80, 90)$ (in order to see better details). We observe a periodic oscillations with the period approximately equal to $\Delta t = 0.7$. Figures 3 and 4 show the Mach number isolines at times $t_i = 89.4 + i/7\Delta t,\ i = 1, 2, \ldots 7$ demonstrating the periodic propagation of vortexes behind profile. These results

are in good agreement with the results from [21] and [9] and moreover, the computational time was shorter in comparison with [9].

It may be surprising that the "weak" stopping condition (26) with $\delta = 0.5$ works also for unsteady flows, particularly any lost of the accuracy was not observed. This is caused by the fact that τ_k is significantly smaller than for steady-state problems and therefore there is a small difference between $\boldsymbol{\xi}_{k-1}$ and $\boldsymbol{\xi}_k$. Hence, the initial residuum at the right-hand side of (26) is already small. This is a large advantage of our approach.

6 Conclusion

We dealt with the semi-implicit discontinuous Galerkin method for the numerical solution of viscous compressible flows which leads to the solution of the sequence of linear algebraic systems. We presented the efficient solution strategy for steady-state flow regimes when the solution of these systems requires shorter computational time than their preparation. This approach requires a choice of two parameters (tolerance ω and δ). However, their choice is very simple. We demonstrate the approach robustness with respect to the mesh size (number of degree of freedom), degree of polynomial approximations and flow regimes.

REFERENCES

[1] F. Bassi and S. Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations. *J. Comput. Phys.*, 131:267–279, 1997.

[2] F. Bassi and S. Rebay. A high order discontinuous Galerkin method for compressible turbulent flow. In B. Cockburn, G. E. Karniadakis, and C.-W. Shu, editors, *Discontinuous Galerkin Method: Theory, Computations and Applications*, Lecture Notes in Computational Science and Engineering 11, pages 113–123. Springer-Verlag, 2000.

[3] C. E. Baumann and J. T. Oden. A discontinuous *hp* finite element method for the Euler and Navier-Stokes equations. *Int. J. Numer. Methods Fluids*, 31(1):79–95, 1999.

[4] B. Cockburn, S. Hou, and C. W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element for conservation laws IV: The multi-dimensional case. *Math. Comp.*, 54:545–581, 1990.

[5] R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact newton methods. *SIAM J. Numer. Anal.*, 19:400–408, 1982.

[6] L. T. Diosady and D. L. Darmofal. Preconditioning methods for discontinuous Galerkin solutions of the Navier-Stokes equations. *J. Comput. Phys.*, 228:3917–3935, 2009.

[7] V. Dolejší. *ANGENER – software package.* Charles University Prague, Faculty of Mathematics and Physics, version 3.0 edition, 2000. webpage: http://www.karlin.mff.cuni.cz/dolejsi/angen.html.

[8] V. Dolejší. On the discontinuous Galerkin method for the numerical solution of the Navier–Stokes equations. *Int. J. Numer. Methods Fluids*, 45:1083–1106, 2004.

[9] V. Dolejší. Semi-implicit interior penalty discontinuous Galerkin methods for viscous compressible flows. *Commun. Comput. Phys.*, 4(2):231–274, 2008.

[10] V. Dolejší. Discontinuous Galerkin method for the numerical simulation of unsteady compressible flow. *WSEAS Transactions on Systems*, 5(5):1083–1090, 2006.

[11] V. Dolejší and M. Feistauer. Semi-implicit discontinuous Galerkin finite element method for the numerical solution of inviscid compressible flow. *J. Comput. Phys.*, 198(2):727–746, 2004.

[12] M. Dumbser and C.-D. Munz. Building blocks for arbitrary high-order discontinuous Galerkin methods. *J. Sci. Comput.*, 27:215–230, 2006.

[13] S. C. Eisenstat and H.F. Walker. Choosing the forcing terms in inexact newton method. *SIAM J. Sci. Comput.*, 17(1):16–32, 1996.

[14] M. Feistauer, J. Felcman, and I. Straškraba. *Mathematical and Computational Methods for Compressible Flow.* Oxford University Press, Oxford, 2003.

[15] M. Feistauer and V. Kučera. On a robust discontinuous galerkin technique for the solution of compressible flow. *J. Comput. Phys.*, 224(1):208–221, 2007.

[16] M. Feistauer, V. Kučera, and J. Prokopová. Discontinuous Galerkin solution of compressible flow in time dependent domains. *Mathematics and Computers in Simulations*, 2009 (in press). DOI: 10.1016/j.matcom.2009.01.020.

[17] R. Hartmann and P. Houston. Symmetric interior penalty DG methods for the compressible Navier-Stokes equations I: Method formulation. *Int. J. Numer. Anal. Model.*, 1:1–20, 2006.

[18] C. M. Klaij, J.J.W. van der Vegt, and H. Van der Ven. Pseudo-time stepping for space-time discontinuous Galerkin discretizations of the compressible Navier-Stokes equations. *J. Comput. Phys.*, 219(2):622–643, 2006.

[19] N. Kroll, H. Bieler, H. Deconinck, V. Couallier, H. van der Ven, and K. Sorensen, editors. *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 Verlag, 2010 (in press).

[20] F. Lörcher, G. Gassner, and C.-D. Munz. A discontinuous Galerkin scheme based on a spacetime expansion. I. Inviscid compressible flow in one space dimension. *J. Sci. Comput.*, 32(2):175–199, 2007.

[21] S. Mittal. Finite element computation of unsteady viscous compressible flows. *Comput. Methods Appl. Mech. Eng.*, 157:151–175, 1998.

[22] 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.

[23] G. Vijayasundaram. Transonic flow simulation using upstream centered scheme of Godunov type in finite elements. *J. Comput. Phys.*, 63:416–433, 1986.

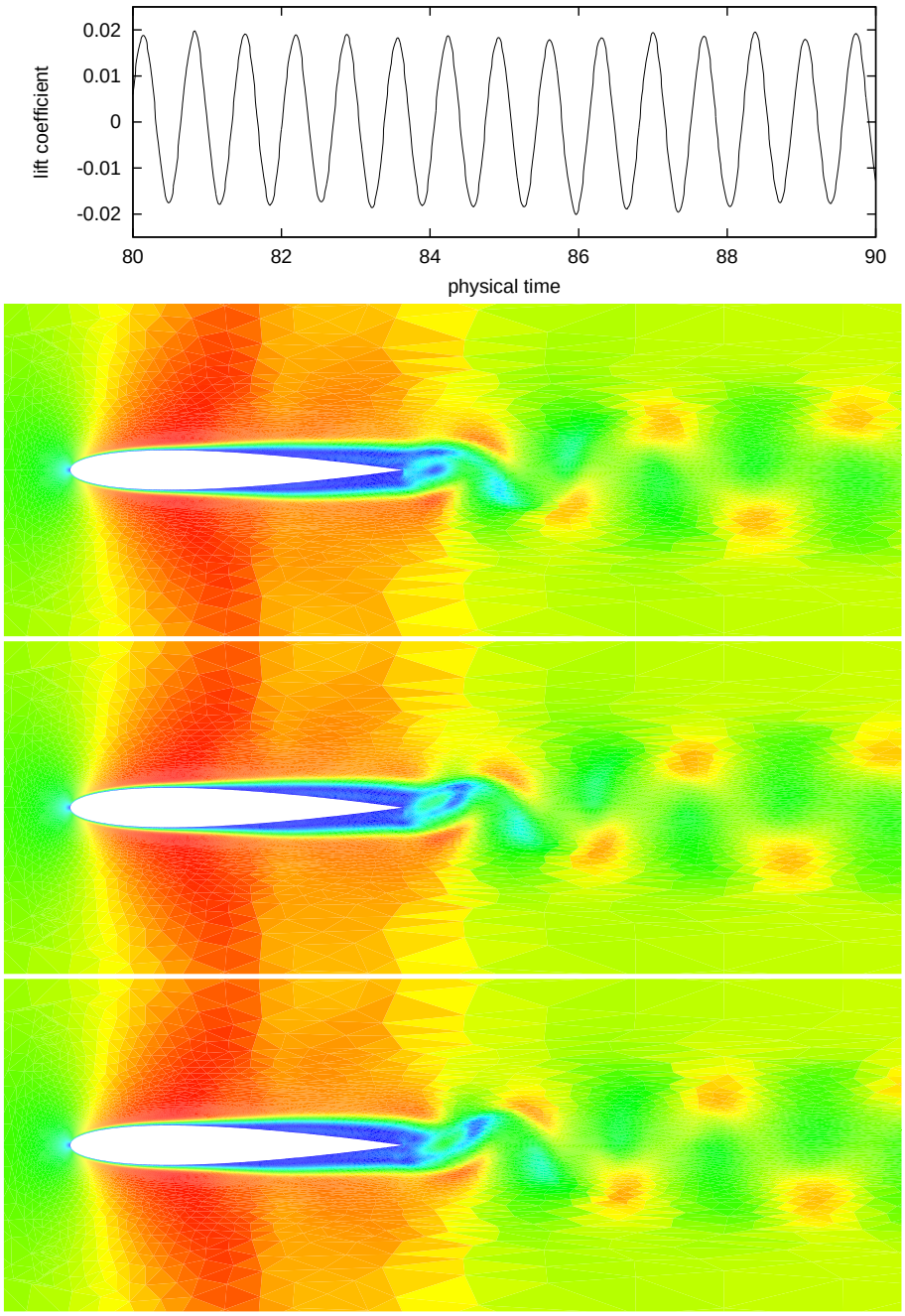

Figure 3: NACA 0012, $M_{\text{inlet}} = 0.85$, $\alpha = 0°$ and $\text{Re} = 10\,000$, dependence of the lift coefficient c_L on time t during $t = (80, 90)$ and the isolines of Mach number at 1/7, 2/7 and 3/7 of one period

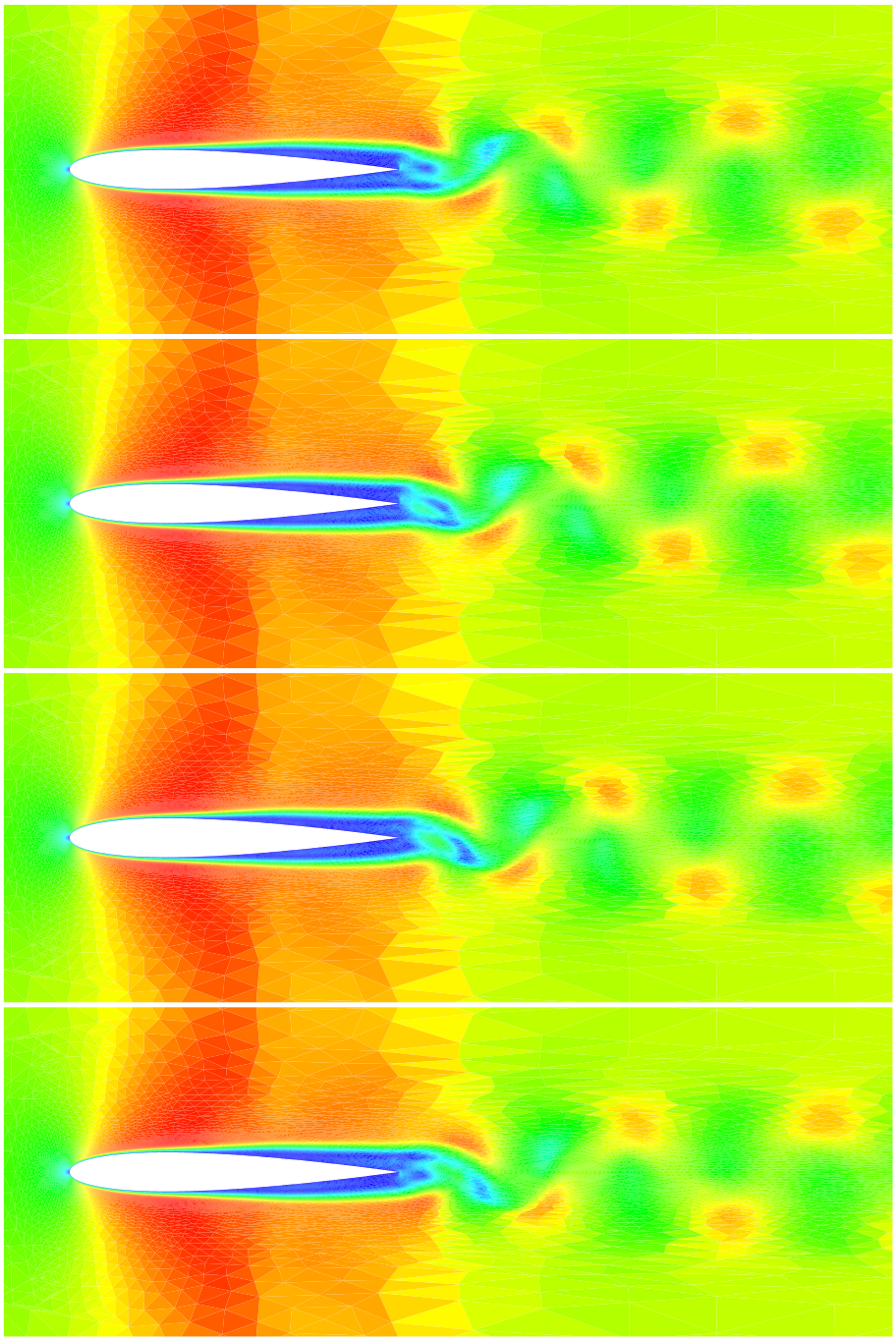

Figure 4: NACA 0012, $M_{\text{inlet}} = 0.85$, $\alpha = 0°$ and $\text{Re} = 10\,000$, the isolines of Mach number at 4/7, 5/7, 6/7 and 7/7 of one period