KYBERNETIKA — VOLUME *46* (2010), NUMBER 2, PAGES 260–280

ON THE SOLUTION OF LINEAR ALGEBRAIC SYSTEMS ARISING FROM THE SEMI–IMPLICIT DGFE DISCRETIZATION OF THE COMPRESSIBLE NAVIER–STOKES EQUATIONS

Vít Dolejší

We deal with the numerical simulation of a motion of viscous compressible fluids. We discretize the governing Navier–Stokes equations by the backward difference formula – discontinuous Galerkin finite element (BDF-DGFE) method, which exhibits a sufficiently stable, efficient and accurate numerical scheme. The BDF-DGFE method requires a solution of one linear algebra system at each time step. In this paper, we deal with these linear algebra systems with the aid of an iterative solver. We discuss the choice of the preconditioner, stopping criterion and the choice of the time step and propose a new strategy which leads to an efficient and accurate numerical scheme.

Keywords: discontinuous Galerkin method, compressible Navier–Stokes equations, linear algebra problems, preconditioning, stopping criterion, choice of the time step

Classification: 76M10, 76N15, 35Q35, 65L06

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 method* (DGM) was employed in many papers for the discretization of compressible fluid flow problems, see, e. g., [3, 4, 5, 9, 10, 14, 16, 17, 19, 20] and the references cited therein. DGM is based on a piecewise polynomial but discontinuous approximation which provides robust and high-order accurate approximations, particularly in transport dominated regimes. Moreover, there is considerable flexibility in the choice of the mesh design; indeed, DGM easily handles non-matching and non-uniform grids, even anisotropic, with different polynomial approximation degrees. This allows a simple treatment of *hp*-variants of adaptive techniques. Additionally, orthogonal bases can easily be constructed which lead to diagonal mass matrices; this is particularly advantageous for unsteady problems. Finally, in combination with block-type preconditioners, DGMs can easily be parallelized.

There are several variants of the DGM for the solution of problems containing diffusion terms, see, e. g., [2]. We employ the *interior penalty Galerkin* (IPG) methods, namely the *symmetric interior penalty Galerkin* (SIPG), the *non-symmetric interior penalty Galerkin* (NIPG) and the *incomplete interior penalty Galerkin* (IIPG) introduced in [1, 22] and [8], respectively.

For unsteady problems, it is possible to use a discontinuous approximation also for the time discretization (e. g., [20, 21]) 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 [3, 5, 7, 9]. 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., [4, 19]. However, a fully implicit scheme leads to a necessity to solve a nonlinear system of algebraic equations at each time step which is rather expensive. Therefore, in [10, 13], 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 problem at each time step. We call this approach the *backward difference formula – discontinuous Galerkin finite element* (BDF-DGFE) method.

The BDF-DGFE method leads to a sequence of linear algebraic problems which should be solved by a suitable solver. Numerical experiments presented in [10] showed that the solution of linear algebra problem consume almost 99% of computational time. Therefore, a significant reduction of computational time necessary for the solution of these problems is a necessary condition for a practical employment of the BDF-DGFE method. Moreover, the amount of the used computer memory has to be taken into account. In this paper, we develop an efficient solution strategy for the mentioned algebraic problems, namely we deal with the choice of preconditioner, stopping criterion and the size of the time step.

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 [10]. In Section 4, we discuss numerical solution of the arising linear algebraic systems, propose an "optimal" strategy and demonstrate its efficiency and accuracy. The concluding remarks are given in Section 5.

2. COMPRESSIBLE FLOW PROBLEM

Let $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, be a bounded domain 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 viscous compressible fluids 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 $\boldsymbol{w} = (\rho, \rho v_1, \ldots, \rho v_d, e)^{\mathrm{T}}$ is the *state vector*, $\boldsymbol{f}_s : \mathbb{R}^{d+2} \to \mathbb{R}^{d+2}$, $s = 1, \ldots, d$, are the inviscid (Euler) fluxes and $\boldsymbol{K}_{sk} : \mathbb{R}^{d+2} \to \mathbb{R}^{(d+2)\times(d+2)}$, $s, k = 1, \ldots, d$, represent the viscous terms. The forms of vectors $\boldsymbol{f}_s$, $s = 1, \ldots, d$, and matrices

$\boldsymbol{K}_{sk}$, $s = 1, \dots, d$, can be found, e.g., in [10] or [15, Section 4.3]. We use the following notation: ρ – density, p – pressure, e – total energy, $\boldsymbol{v} = (v_1, \dots, v_d)$ – velocity, Re – Reynolds number. The system (1) is of *hyperbolic-parabolic* type and it is equipped with a suitable initial and boundary conditions, see [9], [10]. 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).

Let us mention that the Euler fluxes $\boldsymbol{f}_s$, $s = 1, \dots, d$, satisfy (see [15, Lemma 3.1]) $\boldsymbol{f}_s(\boldsymbol{w}) = \boldsymbol{A}_s(\boldsymbol{w})\boldsymbol{w}$, $s = 1, \dots, d$, where $\boldsymbol{A}_s(\boldsymbol{w}) = \frac{D\boldsymbol{f}_s(\boldsymbol{w})}{D\boldsymbol{w}}$, $s = 1, \dots, d$, are the Jacobi matrices of the mappings $\boldsymbol{f}_s$. Finally, we define the matrix

$$\boldsymbol{P}(\boldsymbol{w}, \boldsymbol{n}) = \sum_{s=1}^{d} \boldsymbol{A}_s(\boldsymbol{w}) n_s, \tag{2}$$

where $\boldsymbol{n} = (n_1, \dots, n_d) \in \mathbb{R}^d$, $|n|^2 = n_1^2 + \dots + n_d^2 = 1$, which plays a role in the definition of a numerical flux.

3. DGFE DISCRETIZATION

3.1. Triangulations

Let $\mathcal{T}_h$ $(h > 0)$ be a partition of the domain Ω into a finite number of closed d-dimensional simplexes K with mutually disjoint interiors. I.e., $\overline{\Omega} = \bigcup_{K \in \mathcal{T}_h} K$. We call $\mathcal{T}_h = \{K\}_{K \in \mathcal{T}_h}$ a *triangulation* of Ω and do not require the conforming properties from the finite element method, see, e.g, [6].

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, let $\mathcal{F}_h^I$ be the set of all $\Gamma \in \mathcal{F}_h$ that are contained in Ω (inner faces). Moreover, we denote by $\mathcal{F}_h^w$, $\mathcal{F}_h^i$ and $\mathcal{F}_h^o$ the set 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, let $\mathcal{F}_h^D$ be 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 type boundary conditions are prescribed (i.e., $\mathcal{F}_h^N = \mathcal{F}_h^o$). Obviously, $\mathcal{F}_h = \mathcal{F}_h^I \cup \mathcal{F}_h^D \cup \mathcal{F}_h^N$. 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 edge.

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 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 of vector-valued functions

$$\boldsymbol{S}_{h\mathsf{p}} = S_{h\mathsf{p}} \times \cdots \times S_{h\mathsf{p}} \quad (d+2 \text{ times}). \tag{4}$$

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 $v \in H^1(\Omega, \mathcal{T}_h)$, 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 [10].

3.3.1. Inviscid terms

For $\boldsymbol{w}_h, \bar{\boldsymbol{w}}_h, \boldsymbol{\varphi}_h \in \boldsymbol{S}_{h\mathsf{p}}$, we define the forms

$$\begin{aligned}
\boldsymbol{b}_h(\bar{\boldsymbol{w}}_h, \boldsymbol{w}_h, \boldsymbol{\varphi}_h) = & -\sum_{K \in \mathcal{T}_h} \int_K \sum_{s=1}^d \boldsymbol{A}_s(\bar{\boldsymbol{w}}_h)\boldsymbol{w}_h \cdot \frac{\partial \boldsymbol{\varphi}_h}{\partial x_s} \, \mathrm{d}x \qquad (5)\\
& + \sum_{\Gamma \in \mathcal{F}_h^I} \int_\Gamma \left(\boldsymbol{P}^+ \left(\langle \bar{\boldsymbol{w}}_h \rangle, \boldsymbol{n}\right) \boldsymbol{w}_h|_\Gamma^{(p)} + \boldsymbol{P}^- \left(\langle \boldsymbol{w}_h \rangle, \boldsymbol{n}\right) \boldsymbol{w}_h|_\Gamma^{(n)} \right) \cdot [\boldsymbol{\varphi}_h] \mathrm{d}S \\
& + \sum_{\Gamma \in \mathcal{F}_h^{io}} \int_\Gamma \left(\boldsymbol{P}^+ \left(\langle \bar{\boldsymbol{w}}_h \rangle, \boldsymbol{n}\right) \boldsymbol{w}_h|_\Gamma^{(p)} \right) \cdot [\boldsymbol{\varphi}_h] \, \mathrm{d}S + \sum_{\Gamma \in \mathcal{F}_h^w} \int_\Gamma \boldsymbol{F}_W(\bar{\boldsymbol{w}}_h, \boldsymbol{w}_h, \boldsymbol{n}) \cdot \boldsymbol{\varphi}_h \, \mathrm{d}S, \\
\tilde{\boldsymbol{b}}_h(\bar{\boldsymbol{w}}_h, \boldsymbol{\varphi}_h) = & -\sum_{\Gamma \in \mathcal{F}_h^{io}} \int_\Gamma \left(\boldsymbol{P}^- \left(\langle \bar{\boldsymbol{w}}_h \rangle, \boldsymbol{n}\right) \bar{\boldsymbol{w}}_h|_\Gamma^{(n)} \right) \cdot [\boldsymbol{\varphi}_h] \, \mathrm{d}S,
\end{aligned}$$

where $\boldsymbol{A}_s(\cdot) = 1, \ldots, d$ are the Jacobi matrices of the mappings $\boldsymbol{f}_s$, $s = 1, \ldots, d$, $\boldsymbol{P}^\pm(\cdot, \cdot)$ are the positive and negative parts of the matrix $\boldsymbol{P}(\cdot, \cdot)$ given by (2) which define the Vijayasundaram numerical flux used for the approximation of inviscid fluxes though $\Gamma \in \mathcal{F}_h$. This numerical flux is suitable for the semi-implicit time discretization. Moreover,

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

where $\boldsymbol{DF}_W(\boldsymbol{w},\boldsymbol{n})$ is a $(d+2)\times(d+2)$ matrix obtained by the differentiation of $\sum_{s=1}^{d}\boldsymbol{f}_s(\boldsymbol{w})n_s$ with respect to $\boldsymbol{w}$, see [9], [10] or [13].

Finally, $\bar{\boldsymbol{w}}|_{\Gamma}^{(n)} = LRP(\bar{\boldsymbol{w}}|_{\Gamma}^{(p)},\boldsymbol{w}_D,\boldsymbol{n}_\Gamma)$, $\Gamma\in\mathcal{F}_h^{io}$ 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 [12]. For more details, we refer to [13].

3.3.2. Viscous terms

For $\bar{\boldsymbol{w}}_h,\boldsymbol{w}_h,\boldsymbol{\varphi}_h\in\boldsymbol{S}_{h\mathsf{p}}$, we define the forms

$$\begin{aligned}
\boldsymbol{a}_h(\bar{\boldsymbol{w}}_h,\boldsymbol{w}_h,\boldsymbol{\varphi}_h) \;=\;& \sum_{K\in\mathcal{T}_h}\int_K \sum_{s,k=1}^{d}\left(\boldsymbol{K}_{s,k}(\bar{\boldsymbol{w}}_h)\frac{\partial\boldsymbol{w}_h}{\partial x_k}\right)\cdot\frac{\partial\boldsymbol{\varphi}_h}{\partial x_s}\,\mathrm{d}x \qquad (7)\\
&-\sum_{\Gamma\in\mathcal{F}_h^{ID}}\int_\Gamma\sum_{s=1}^{d}\left\langle\sum_{k=1}^{d}\boldsymbol{K}_{s,k}(\bar{\boldsymbol{w}}_h)\frac{\partial\boldsymbol{w}_h}{\partial x_k}\right\rangle n_s\cdot[\boldsymbol{\varphi}_h]\,\mathrm{d}S\\
&-\eta\sum_{\Gamma\in\mathcal{F}_h^{ID}}\int_\Gamma\sum_{s=1}^{d}\left\langle\sum_{k=1}^{d}\boldsymbol{K}_{s,k}^{\mathrm{T}}(\bar{\boldsymbol{w}}_h)\frac{\partial\boldsymbol{\varphi}_h}{\partial x_k}\right\rangle n_s\cdot[\boldsymbol{w}_h]\,\mathrm{d}S,\\
\tilde{\boldsymbol{a}}_h(\bar{\boldsymbol{w}}_h,\boldsymbol{\varphi}_h) \;=\;& -\eta\sum_{\Gamma\in\mathcal{F}_h^{D}}\int_\Gamma\sum_{s,k=1}^{d}\boldsymbol{K}_{s,k}^{\mathrm{T}}(\bar{\boldsymbol{w}}_h)\frac{\partial\boldsymbol{\varphi}_h}{\partial x_k}n_s\cdot\boldsymbol{w}_B\,\mathrm{d}S,
\end{aligned}$$

The state vector $\boldsymbol{w}_B$ prescribed on $\partial\Omega_i\cup\partial\Omega_w$ is given by the boundary conditions, see [9] or [10].

The value of η appearing in (7) can be chosen arbitrarily but the most usual are the values $-1,0$ and 1. Then we obtain three variants of the DGFE scheme:

$\eta=1$ – *symmetric interior penalty Galerkin* (SIPG),

$\eta=-1$ – *non-symmetric interior penalty Galerkin* (NIPG),

$\eta=0$ – *incomplete interior penalty Galerkin* (IIPG).

The numerical analysis of these variants applied to the Poisson equation was presented in [2]. A numerical study of these variants applied to the Navier–Stokes equations was given in [10].

3.3.3. Interior and boundary penalties

For $\boldsymbol{w}_h,\boldsymbol{\varphi}_h\in\boldsymbol{S}_{h\mathsf{p}}$, we define the forms

$$\boldsymbol{J}_h^{\sigma}(\boldsymbol{w},\boldsymbol{\varphi})=\sum_{\Gamma\in\mathcal{F}_h^{ID}}\int_\Gamma\sigma[\boldsymbol{w}]\cdot[\boldsymbol{\varphi}]\,\mathrm{d}S,\qquad \tilde{\boldsymbol{J}}_h^{\sigma}(\boldsymbol{\varphi})=\sum_{\Gamma\in\mathcal{F}_h^{D}}\int_\Gamma\sigma\boldsymbol{w}_B\cdot\boldsymbol{\varphi}\,\mathrm{d}S, \qquad (8)$$

where $\boldsymbol{w}_B$ is the boundary state and the penalty parameter σ is chosen by

$$\sigma|_\Gamma = C_W(\mathrm{diam}(\Gamma)\,\mathrm{Re})^{-1},\quad \Gamma\in\mathcal{F}_h^{ID}, \qquad (9)$$

where $C_W > 0$ 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 [10] where a numerical study was presented.

3.3.4. Semi-implicit BDF-DGFE discretization

In order to simplify the notation, for $\bar{\boldsymbol{w}}_h$, $\boldsymbol{w}_h$, $\boldsymbol{\varphi}_h \in \boldsymbol{S}_{h\mathsf{p}}$, we put

$$\begin{aligned} \boldsymbol{c}_h\left(\bar{\boldsymbol{w}}_h, \boldsymbol{w}_h, \boldsymbol{\varphi}_h\right) &= \boldsymbol{a}_h\left(\bar{\boldsymbol{w}}_h, \boldsymbol{w}_h, \boldsymbol{\varphi}_h\right) + \boldsymbol{b}_h\left(\bar{\boldsymbol{w}}_h, \boldsymbol{w}_h, \boldsymbol{\varphi}_h\right) + \boldsymbol{J}_h^{\sigma}\left(\boldsymbol{w}_h, \boldsymbol{\varphi}_h\right), \\ \tilde{\boldsymbol{c}}_h\left(\bar{\boldsymbol{w}}_h, \boldsymbol{\varphi}_h\right) &= \tilde{\boldsymbol{a}}_h\left(\bar{\boldsymbol{w}}_h, \boldsymbol{\varphi}_h\right) + \tilde{\boldsymbol{b}}_h\left(\bar{\boldsymbol{w}}_h, \boldsymbol{\varphi}_h\right) + \tilde{\boldsymbol{J}}_h^{\sigma}\left(\boldsymbol{\varphi}_h\right). \end{aligned} \tag{10}$$

It is possible to show (see, e. g., [9], [10]) that if $\boldsymbol{w} : \Omega \times (0, T) \to \mathbb{R}^{d+2}$ is a continuously differentiable function satisfying the Navier–Stokes equations (1) and the corresponding initial and boundary conditions then

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\boldsymbol{w}, \boldsymbol{\varphi}\right) + \boldsymbol{c}_h\left(\boldsymbol{w}, \boldsymbol{w}, \boldsymbol{\varphi}\right) = \tilde{\boldsymbol{c}}_h\left(\boldsymbol{w}, \boldsymbol{\varphi}\right) \quad \forall \boldsymbol{\varphi} \in \boldsymbol{S}_{h\mathsf{p}}. \tag{11}$$

Now, we introduce the *semi-discrete problem.*

Definition 3.1. Function $\boldsymbol{w}_h$ is called the *semi-discrete solution* of (CFP), if

$$\begin{aligned} &\text{a)} && \boldsymbol{w}_h \in C^1([0, T]; \boldsymbol{S}_{h\mathsf{p}}), \\ &\text{b)} && \left(\frac{\partial \boldsymbol{w}_h(t)}{\partial t}, \boldsymbol{\varphi}_h\right) + \boldsymbol{c}_h(\boldsymbol{w}_h(t), \boldsymbol{w}_h(t), \boldsymbol{\varphi}_h) = \tilde{\boldsymbol{c}}_h(\boldsymbol{w}_h(t), \boldsymbol{\varphi}_h) \\ &&& \qquad \forall\, \boldsymbol{\varphi}_h \in \boldsymbol{S}_{h\mathsf{p}}\ \forall\, t \in (0, T), \\ &\text{c)} && \boldsymbol{w}_h(0) = \boldsymbol{w}_h^0, \end{aligned} \tag{12}$$

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 (12), a) – c) represents a system of ordinary differential equations (ODEs) for $\boldsymbol{w}_h(t)$ which has to be discretized in time by a suitable method.

As we already mentioned in Introduction, we discretize the semi-discrete problem (12), a) – c) by a *semi-implicit* technique developed in [10] and [13] . We employ the linearity of the form $\boldsymbol{c}_h(\cdot, \cdot, \cdot)$ with respect to its second argument, which follows from expressions (5), (7), (8) and (10). Hence, in the following section, the time derivative term in (12), b) is approximated by a multi-step formula, the second argument of $\boldsymbol{c}_h(\cdot, \cdot, \cdot)$ is discretized implicitly and the first one by an explicit higher order extrapolation.

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

Definition 3.2. Let $n \geq 1$. We define the *approximate solution* of (CFP) by the n-step BDF-DGFE scheme as functions $\boldsymbol{w}_{h,k},\ k = 1, \dots, r$, satisfying the conditions

$$
\begin{aligned}
&\text{a)} \qquad \boldsymbol{w}_{h,k} \in \boldsymbol{S}_{h\mathsf{p}}, \\
&\text{b)} \qquad \frac{1}{\tau_k}\left(\sum_{l=0}^{n} (\alpha_{n,l}\boldsymbol{w}_{h,k-l})\,,\ \boldsymbol{\varphi}_h\right) + \boldsymbol{c}_h\left(\sum_{l=1}^{n} (\beta_{n,l}\boldsymbol{w}_{h,k-l})\,,\ \boldsymbol{w}_{h,k},\ \boldsymbol{\varphi}_h\right) \\
&\qquad\quad = \tilde{\boldsymbol{c}}_h\left(\sum_{l=1}^{n} (\beta_{n,l}\boldsymbol{w}_{h,k-l})\,,\ \boldsymbol{\varphi}_h\right) \qquad \forall\, \boldsymbol{\varphi}_h \in \boldsymbol{S}_{h\mathsf{p}},\ k = n, \dots, r, \\
&\text{c)} \qquad \boldsymbol{w}_{h,0} \in S_{h\mathsf{p}} \text{ is an approximation of } \boldsymbol{w}^0, \\
&\text{d)} \qquad \boldsymbol{w}_{h,l} \in \boldsymbol{S}_{h\mathsf{p}},\ l = 1, \dots, n-1, \text{ are given by a suitable one-step method,}
\end{aligned}
\tag{13}
$$

where the coefficients $\alpha_{n,l},\ l = 0, \dots, n$, and $\beta_{n,l},\ l = 1, \dots, n$, depend on $\vartheta_{k-l},\ l = 0, \dots, n$, and for $n = 1, 2, 3$, see [11] or [18, Section III.5].

Remark 3.3. The n-step BDF-DGFE scheme (13), a) – d) corresponds to the well known *multi-step formulae*

$$\sum_{l=0}^{n} \alpha_{n,l}\, y_{k-l} = \tau_k F(y_k), \qquad k = n, \dots, r, \tag{14}$$

used for the numerical solution of ordinary differential equations

$$\frac{\mathrm{d}y(t)}{\mathrm{d}t} = F(y), \qquad y(0) = y_0, \tag{15}$$

where $y : (0, T) \to \mathbb{R}^m$, $y_0 \in \mathbb{R}^m$ and $F : (0, T) \times \mathbb{R}^m \to \mathbb{R}^m$ $(m \in \mathbb{N},\ m \geq 1)$. By y_k we denote the approximation of $y(t_k)$ obtained by (14).

Remark 3.4. In the first ("nonlinear") argument of form $\boldsymbol{c}_h$ in (13), b) we employ a higher order explicit extrapolation

$$\boldsymbol{w}_{h,k} \approx \sum_{l=1}^{n} (\beta_{n,l}\boldsymbol{w}_{h,k-l})\,, \qquad k = n, n+1, \dots . \tag{16}$$

This extrapolation avoids a solution of nonlinear algebraic problem at each time step and keeps the accuracy with respect to the time.

Remark 3.5. Problem (13), a) – d) represents a linear algebraic system for each $k = n, \dots, r$ which should be solved by a suitable solver, see Section 4.

Remark 3.6. If the BDF scheme is stable then the resulting BDF-DGFE method is practically unconditionally stable, has a high order of accuracy with respect to the time and space coordinates and at each time step we have to solve only one linear algebra problem.

4. SOLUTION OF LINEAR ALGEBRAIC SYSTEMS

As we already mentioned, problem (13), a) – d) represents a sequence of linear algebraic systems whose sufficiently accurate and efficient solution is a necessary condition for the practical use of the BDF-DGFE scheme. This is a subject of this section.

4.1. Linear algebra representation

In order to describe the corresponding algebraic problems, we introduce an index set $I \subset \mathbb{Z}^+$(= 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 $K_\mu,\ \mu \in I$. Since $\boldsymbol{S}_{h\mathsf{p}}$ is a space of discontinuous piecewise polynomial functions, for each $K_\mu \in \mathcal{T}_h$ it is possible to define a *local basis*

$$\begin{aligned} B_\mu \quad = \quad & \Big\{\boldsymbol{\psi}_{\mu,j};\ \boldsymbol{\psi}_{\mu,j} \in \boldsymbol{S}_{h\mathsf{p}},\ \mathrm{supp}(\boldsymbol{\psi}_{\mu,j}) \subset K_\mu, \\ & \boldsymbol{\psi}_{\mu,j} \text{ are linearly independent for } j = 1,\dots,\mathsf{dof}_\mu\Big\}, \end{aligned} \tag{17}$$

where

$$\mathsf{dof}_\mu = \frac{d+2}{d!}\Pi_{j=1}^{d}(p_\mu + j), \quad \mu \in I, \tag{18}$$

denotes the number of *local degrees of freedom* for each element $K_\mu \in \mathcal{T}_h$ (we recall that (CFP) represents $d+2$ equations). The values dof_μ are shown in Table 1 for $p_\mu = 1,\dots,5$ and $d = 2,3$. For the construction of the basis $B_\mu,\ \mu \in I$, see Section 4.2.

Table 1. Values of dof_μ for $p_\mu = 1,\dots,5$ and $d = 2,3$

p_μ	1	2	3	4	5
$d=2$	12	24	40	60	84
$d=3$	20	50	100	175	280

A composition of the local basis $B_\mu,\ \mu \in I$ defines a basis of $\boldsymbol{S}_{h\mathsf{p}}$, i. e.,

$$B = \{\boldsymbol{\psi}_j;\ \boldsymbol{\psi}_j \in \boldsymbol{S}_{h\mathsf{p}},\ j = 1,\dots,\mathsf{dof}\}. \tag{19}$$

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

$$\boldsymbol{w}_{h,k}(x) = \sum_{\mu\in I}\sum_{j=1}^{\mathsf{dof}_\mu} \xi_{k,\mu,j}\boldsymbol{\psi}_{\mu,j}(x), \quad x \in \Omega,\ k = 0,1,\dots,r, \tag{20}$$

where $\xi_{k,\mu,j} \in \mathbb{R},\ j = 1, \dots, \mathsf{dof}_\mu,\ \mu \in I,\ k = 0, \dots, r$. Moreover, for $\boldsymbol{w}_{h,k} \in \boldsymbol{S}_{h\mathsf{p}}$, we define a vector of its basis coefficients by

$$\boldsymbol{W}_k = \{\xi_{k,\mu,j}\}_{j=1,\dots,\mathsf{dof}_\mu}^{\mu\in I} \in \mathbb{R}^{\mathsf{dof}}, \quad k = 0, 1, \dots, r. \tag{21}$$

Therefore, using (20) – (21) we have an isomorphism

$$\boldsymbol{w}_{h,k} \in \boldsymbol{S}_{h\mathsf{p}} \quad \longleftrightarrow \quad \boldsymbol{W}_k \in \mathbb{R}^{\mathsf{dof}}. \tag{22}$$

Then problem (13) can be written in the matrix form

$$\boldsymbol{A}_k \boldsymbol{W}_k = \boldsymbol{q}_k, \quad \boldsymbol{A}_k := \boldsymbol{M} + \tau_k \boldsymbol{C}_k, \quad k = n, \dots, r, \tag{23}$$

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

$$\boldsymbol{M} = \mathrm{diag}\{\boldsymbol{M}_{\mu,\mu},\ \mu \in I\}, \qquad \boldsymbol{M}_{\mu,\mu} = \{M_\mu^{i,j}\}_{i,j=1}^{\mathsf{dof}_\mu},\ \mu \in I, \tag{24}$$
$$M_\mu^{i,j} = \alpha_{n,0} \int_\Omega \boldsymbol{\psi}_{\mu,i} \cdot \boldsymbol{\psi}_{\mu,j} \,\mathrm{d}x,$$

the matrix $\boldsymbol{C}_k,\ k = 1, 2, \dots$ is the *"flux" matrix* corresponding to form $\boldsymbol{c}_h(\cdot,\cdot,\cdot)$ at t_k defined by

$$\boldsymbol{C}_k \ = \ \{C_{k,(\mu,i),(\nu,j)}\}_{\mu,\nu\in I}^{i=1,\dots,\mathsf{dof}_\mu, j=1,\dots,\mathsf{dof}_\nu}, \tag{25}$$
$$C_{k,(\mu,i),(\nu,j)} = \boldsymbol{c}_h\left(\sum_{l=1}^{n} \beta_{n,l} \boldsymbol{w}_{h,k-l},\ \boldsymbol{\psi}_{\mu,i},\ \boldsymbol{\psi}_{\nu,j}\right),$$

and $\boldsymbol{q}_k \in \mathbb{R}^{\mathsf{dof}}$ represents the right-hand-sides of (13), b) given by

$$\boldsymbol{q}_k \ = \ \{q_{k,\mu,i}\}_{\mu\in I}^{i=1,\dots,\mathsf{dof}_\mu}, \tag{26}$$
$$\{q_{k,\mu,i}\} = -\left(\sum_{l=1}^{n} \alpha_{n,l} \boldsymbol{w}_{h,k-l},\ \boldsymbol{\psi}_{\mu,i}\right) + \tau_k \tilde{\boldsymbol{c}}_h \left(\sum_{l=1}^{n} (\beta_{n,l} \boldsymbol{w}_{h,k-l}),\ \boldsymbol{\psi}_{\mu,i}\right).$$

In virtue of the local character of basis B it is easy to observe that the matrices $\boldsymbol{C}_k,\ k = n, \dots, r$ have a block structure. From the expressions (5), (7), (8) and (10) it follows that the matrix element $C_{k,(\mu,i),(\nu,j)}$ is non-vanishing if $\mu = \nu$ or if elements K_μ and K_ν share an face. The size of a non-diagonal block is equal to $\mathsf{dof}_\mu \times \mathsf{dof}_\nu$ and the number of non-diagonal blocks corresponding to an element $K_\mu \in \mathcal{T}_h$ is equal to the number of neighbouring elements of K_μ. Then we can write the block-structure of $\boldsymbol{C}_k$ by

$$\boldsymbol{C}_k \ = \ \{\boldsymbol{C}_{k,\mu,\nu}\}_{\mu,\nu\in I}^{\partial K_\mu \cap \partial K_\nu \neq \emptyset}, \quad \boldsymbol{C}_{k,\mu,\nu} = \{C_{k,(\mu,i),(\nu,j)}\}_{i=1,\dots,\mathsf{dof}_\mu}^{j=1,\dots,\mathsf{dof}_\nu}, \tag{27}$$

where $\boldsymbol{C}_{k,\mu,\nu}$ represents a $\mathsf{dof}_\mu \times \mathsf{dof}_\nu$-block with elements $C_{k,(\mu,i),(\nu,j)}$ given by (25), $\mu, \nu \in I,\ k = n, \dots, r,$.

4.2. Choice of shape functions

We employ the local character of the shape functions and construct basis of $\boldsymbol{S}_{h\mathsf{p}}$ which is orthonormal with respect to the L^2-scalar product. Let

$$\hat{K} = \{(\hat{x}_1, \dots, \hat{x}_d);\ \hat{x}_i \geq 0,\ i = 1, \dots, d,\ \sum_{i=1}^{d} \hat{x}_i \leq 1\} \tag{28}$$

be a reference simplex, we define a basis of the space of vector-valued polynomials of degree $\leq p$ on $\hat{K}$ by

$$\begin{aligned} \hat{\boldsymbol{S}}_p &= \hat{S}_p \times \cdots \times \hat{S}_p \quad (d+2 \text{ times}), \qquad (29) \\ \hat{S}_p &= \{\phi_{n_1,\dots,n_d}(\hat{x}_1, \dots, \hat{x}_d) = \Pi_{i=1}^{d} (\hat{x}_i - \hat{x}_i^c)^{n_i};\ n_1, \dots, n_d \geq 0,\ \sum_{j=1}^{d} n_j \leq p\}, \end{aligned}$$

where $(\hat{x}_1^c, \dots, \hat{x}_d^c)$ is the barycentre of $\hat{K}$. Obviously, the set $\hat{\boldsymbol{S}}_p$ is a basis of the space of vector-valued polynomials on the reference element $\hat{K}$ of degree $\leq p$. By the Gram–Schmidt orthogonalization process applied to $\hat{\boldsymbol{S}}_p$ we obtain the orthonormal set $\{\hat{\phi}_j,\ j = 1, \dots, \mathsf{dof}_\mu\}$ where dof_μ is given by (18) with $p_\mu := p$.

Furthermore, let $F_\mu,\ \mu \in I$, be a linear mapping of the reference element $\hat{K}$ onto the element K_μ. Then we put

$$B_\mu = \{\boldsymbol{\psi}_{\mu,j},\ \boldsymbol{\psi}_{\mu,j}(x) = \boldsymbol{\psi}_{\mu,j}(F_\mu(\hat{x})) = \hat{\phi}_j(\hat{x}),\ j = 1, \dots, \mathsf{dof}_\mu\}, \tag{30}$$

which define an orthogonal basis B_μ introduced in (17) for each element $K_\mu \in \mathcal{T}_h$ separately. Then blocks $\boldsymbol{M}_{\mu,\mu}$ of the mass matrix $\boldsymbol{M}$ given by (24) are diagonal. Finally, (19) defines the orthogonal basis of $\boldsymbol{S}_{h\mathsf{p}}$. The Gram–Schmidt orthogonalization on the reference element can be carried out by a symbolical computing, hence the orthogonalization does not cause any loss of accuracy.

If we consider curved elements approximating nonpolygonal boundaries then the mapping $F_\mu : \hat{K} \to K_\mu$ is not linear and the transformation of the basis in (30) slightly violates the orthogonality. Moreover, the matrix block $\boldsymbol{M}_{\mu,\mu}$ is not diagonal but fortunately strongly diagonally dominant.

4.3. Solution of the sequence of the linear algebraic problems (23)

The sequence of the linear algebraic problems (23) has to be numerically solved at each time level $t_k,\ k = n, \dots, r$. It is possible to use a direct solver which is efficient for not very large matrix (usually $\mathsf{dof} \approx 10^4 - 10^5$). For larger systems, it is suitable to use some iterative solver. We employ the restarted GMRES solver with a suitable preconditioning which is a widely used technique for the solution of nonsymmetric linear algebraic systems.

Therefore, at each time level $t_k,\ k = n, \dots, r$, we solve the problem (23) approximately and instead of $\boldsymbol{W}_k,\ k = n, \dots, r$, we obtain their approximations $\bar{\boldsymbol{W}}_k$.

Formally, we define the GMRES iterative process at the k^{th} level, $k = n, \dots, r$, by

$$\begin{aligned} &\text{i)} && \bar{\boldsymbol{W}}_k^0 := \bar{\boldsymbol{W}}_{k-1}, \\ &\text{ii)} && \bar{\boldsymbol{W}}_k^l := \mathsf{GMRES\_iter}(\boldsymbol{A}_k, \boldsymbol{q}_k, \bar{\boldsymbol{W}}_k^{l-1}), \quad l = 1, \dots, s_k, \\ &\text{iii)} && \bar{\boldsymbol{W}}_k := \bar{\boldsymbol{W}}_k^{s_k}, \end{aligned} \tag{31}$$

where $\mathsf{GMRES\_iter}$ formally denotes one step of GMRES method for a given matrix, right-hand side and initial vector and s_k, $k = n, \dots, r$, is the number of inner GMRES steps (iterations), which should be chosen on the basis of a suitable stopping criterion.

In order to develop a sufficiently accurate and efficient numerical scheme, we deal with the following three aspects,

– *choice of the preconditioner*,

– *choice of the stopping criterion*,

– *choice of the time step*,

which significantly influence the accuracy and efficiency of the method. All these aspects are not independent and therefore they have to be considered together.

4.3.1. Choice of the preconditioner

If the matrix $\boldsymbol{A}_k$, $k = n, \dots, r$, in (23) is ill-conditioned, then the GMRES iterative process requires many iterations in order to achieve a given accuracy. Therefore, it is convenient to apply a suitable preconditioner $\hat{\boldsymbol{P}}_k \in \mathbb{R}^{\mathsf{dof}\times\mathsf{dof}}$ to problem (23) which leads to the problem

$$\hat{\boldsymbol{P}}_k \boldsymbol{A}_k \boldsymbol{W}_k = \hat{\boldsymbol{P}}_k \boldsymbol{q}_k, \quad k = n, \dots, r, \tag{32}$$

equivalent to (23). Suitable preconditioner means that the condition number of the matrix $\hat{\boldsymbol{P}}_k \boldsymbol{A}_k$ is significantly smaller than the condition number of $\boldsymbol{A}_k$.

There exist many various preconditioners for the solution of linear systems. For the efficiency of the BDF-DGFEM scheme we require

- a *high efficiency of the preconditioner*, which means that it significantly reduces the number of iterations of the iterative (GMRES) solver,
- a *low costs of the preconditioner*, which means that computational time for the evaluation of the preconditioner and the used memory are low.

We will mention two basic preconditioners. The simplest one is the *block diagonal preconditioner* (BDP) when $\hat{\boldsymbol{P}}_k = \mathrm{diag}\{\hat{\boldsymbol{P}}_{k,\mu,\mu},\ \mu \in I\}$ has the same block structure as the mass matrix $\boldsymbol{M}$ and their blocks are given by $\hat{\boldsymbol{P}}_{k,\mu,\mu} := (\boldsymbol{M}_{\mu,\mu} + \tau_k \boldsymbol{C}_{k,\mu,\mu})^{-1}$, $k = n, \dots, r$. The inversions of diagonal blocks are evaluated directly by elimination. Since these blocks are relatively small (see Table 1) the evaluation of BDP is fast. Moreover, BDP requires a small amount of additional memory for storing of $\hat{\boldsymbol{P}}_k$, $k = n, \dots, r$, the same memory as the storing of $\boldsymbol{M}$.

Another possibility is the *block ILU(0) preconditioner*, which belongs among the incomplete LU preconditioners, where the matrix $\boldsymbol{A}_k$ is decomposed by an incomplete LU algorithm (see [23]) such that the block sparsity of the preconditioner is the same as the original matrices $\boldsymbol{A}_k,\ k = n, \dots, r$. We only mention that the computational time for (incomplete) LU decomposition is significantly large then the computational time for BDP. The use of ILU preconditioner is the subject of further research.

In the following, BDP will be considered. We can expect that the efficiency of BDP depends on the size of the time step τ_k. Namely for small τ_k the efficiency is high since it follows from (23) that the diagonal blocks of $\boldsymbol{A}_k,\ k = n, \dots, r$, dominate the off-diagonal ones. On the other hand, this dominance is decreasing for increasing τ_k and consequently the efficiency of BDP becomes smaller. This expectation can be demonstrated by the following example.

Example 1. We consider a linear algebraic system

$$(\boldsymbol{M} + \tau \boldsymbol{C})\, \boldsymbol{W} = \boldsymbol{q}, \qquad \tau > 0, \tag{33}$$

where $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{q}$ are given (fixed) mass matrix, flux matrix and right-hand side, respectively. The size of problem (33) is $\mathsf{dof} = 121\,000$ and the matrix $\boldsymbol{C}$ has 19 147 200 nonzero elements. Matrices $\boldsymbol{M}$, $\boldsymbol{C}$ and vector $\boldsymbol{q}$ correspond to a real CFP. We carried out several solutions of (33) by GMRES with BDP for different sizes of time step, from $\tau = 10^{-5}$ to $\tau = 0.5$. We emphasize at only one time step (with different size) was carried out for each computation.

Figure 1 shows the dependence of the number of GMRES iterations (left) and the total computational time (right) in seconds on τ. As the stopping criterion condition (36) with $\omega = 10^{-6}$ was used. We observe an exponential increase of computational time in dependence on τ.

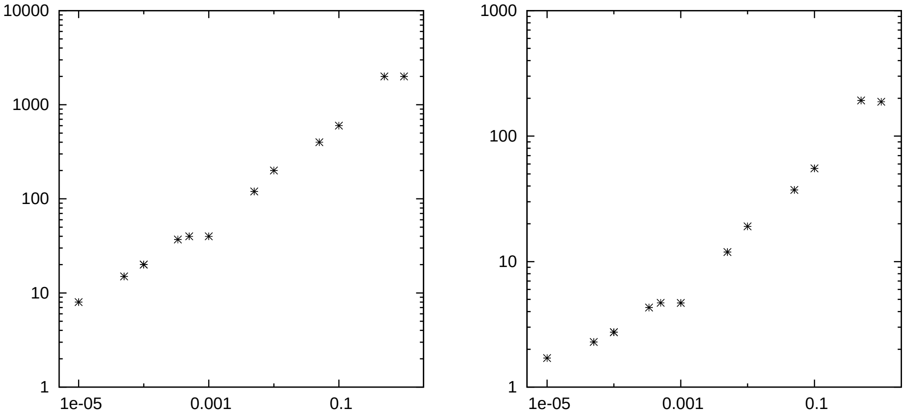

Fig. 1. Dependence of the number of iterations (left) and the computational time (right) on τ.

Moreover, numerical experiments show that the computational time for the calculation of the flux matrix $\boldsymbol{C}$ is approximately equal to 35 GMRES iterations for P_1 approximation, 40 GMRES iterations for P_2 approximation and 50 GMRES iterations for P_3 approximation. Therefore, we can conclude that BDP is sufficiently efficient for the time step up to $\tau \approx 10^{-3}$. A general strategy for the choice of the time step is discussed in Section 4.3.3.

4.3.2. Stopping criterion

The choice of the stopping criterion for the iterative solver (31) is fundamental in the context of the accuracy and efficiency of the BDF-DGFE method. A rather weak stopping criterion can cause a loss of accuracy and on the other hand too strong condition leads to a significant increase of the computational time.

Abstract optimal strategy is to stop (31) when additional iterations do not reduce (essentially) computational error. We have to take into account two types of computational errors:

- *discretization error* given by $\boldsymbol{e}_D^k := \boldsymbol{w}_{h,k} - \boldsymbol{w}(\cdot, t_k), \quad k = n, \dots, r,$ where $\boldsymbol{w}_{h,k}(x)$ is the approximate solution (13) evaluated in the exact arithmetic and $\boldsymbol{w}(\cdot, t_k)$ is the exact solution of (1) at time $t_k,\ k = n, \dots, r,$
- *algebraic error* given by $\boldsymbol{e}_A^k := \bar{\boldsymbol{w}}_{h,k} - \boldsymbol{w}_{h,k}$, where $\bar{\boldsymbol{w}}_{h,k} \in \boldsymbol{S}_{h\mathsf{p}}$ is a function corresponding to $\bar{\boldsymbol{W}}_k$ (resulting from (31)) given by the isomorphism (22).

In order to balance between *efficiency* and *accuracy* of the BDF-DGFE method, it is natural to stop the algorithm if

$$\|\boldsymbol{e}_A\| \le \|\boldsymbol{e}_D\|, \tag{34}$$

where $\|\cdot\|$ denotes a suitable norm.

However, the principal obstacle in the practical use of (34) is the impossibility to evaluate $\boldsymbol{e}_D^k$. Even in case when some a posteriori estimates of the error are available, we can employ only the approximate solution $\bar{\boldsymbol{w}}_{h,k}$ which is influenced from the algebraic error. Moreover, an evaluation of algebraic error is problematic in general case.

The impossibility of the evaluation or approximation of the algebraic error leads to the use of simpler techniques for the choice of the stopping criterion of (31), namely various residuum-type criteria. The simplest one is the *residuum criterion* in the form

$$\mathrm{res}_k := \|\boldsymbol{A}_k \bar{\boldsymbol{W}}_k^{s_k} - \boldsymbol{q}_k\| \le \omega_1,\ k = n, \dots, r, \tag{35}$$

where $\omega_1 > 0$ is a given tolerance. The residuum res_k is related to the algebraic error by $\|\boldsymbol{e}_A^k\| = \|\bar{\boldsymbol{W}}_k - \boldsymbol{W}_k\| \le \|\boldsymbol{A}_k^{-1}\|\mathrm{res}_k$. However, condition (35) is problematic for ill-conditioned matrices $\boldsymbol{A}_k$ (which is our case) since $\|\boldsymbol{A}_k^{-1}\| \gg 1$.

Therefore, more sophisticated approach is the use of the *preconditioned residuum criterion* in the form

$$P\mathrm{res}_k := \|\hat{\boldsymbol{P}}_k \boldsymbol{A}_k \bar{\boldsymbol{W}}_k^{s_k} - \hat{\boldsymbol{P}}_k \boldsymbol{q}_k\| \le \omega,\ k = n, \dots, r, \tag{36}$$

where $\hat{\boldsymbol{P}}_k$ is the preconditioning matrix and $\omega > 0$ is a given tolerance. This stopping criterion reflects the algebraic error much better in case when $\hat{\boldsymbol{P}}_k \approx \boldsymbol{A}_k^{-1}$, i. e., when the preconditioner is efficient. However, this is not the case of BDP with large τ_k as was mentioned in Section 4.3.1.

Hence, we propose the so-called *difference criterion* in the way that the iterative process (31) is stopped if

$$Dres_k := \frac{\|\bar{\boldsymbol{W}}_k^{s_k} - \bar{\boldsymbol{W}}_k^{s_k-1}\|}{\|\bar{\boldsymbol{W}}_k^{s_k}\|} \le \omega_2, \;\; k = n, \dots, r, \tag{37}$$

where $\omega_2 > 0$ is a given tolerance. In practical computation we use the value $\omega_2 = 10^{-6}$. The proposed difference criterion may be problematic in some situations, when the difference (37) may be small but the algebraic error is still high. Nevertheless, numerical experiments show that there is not any loss of the accuracy at least for the steady state problems.

Remark 4.1. If the weighted discrete ℓ^2-norm $\|\boldsymbol{W}_k\|^2_{w\ell^2} := \sum_{\mu\in I} |K_\mu| \sum_{j=1}^{\mathsf{dof}_\mu} \xi^2_{k,\mu,j}$ is considered in (37) and the basis B is orthogonal then

$$\frac{\|\bar{\boldsymbol{W}}_k^{s_k} - \bar{\boldsymbol{W}}_k^{s_k-1}\|_{w\ell^2}}{\|\bar{\boldsymbol{W}}_k^{s_k}\|_{w\ell^2}} = \frac{\|\bar{\boldsymbol{w}}_h^{k,s_k} - \bar{\boldsymbol{w}}_h^{k,s_k-1}\|_{L^2(\Omega)}}{\|\bar{\boldsymbol{w}}_h^{k,s_k}\|_{L^2(\Omega)}}, \tag{38}$$

where $\bar{\boldsymbol{w}}_h^{k,s_k}$ and $\bar{\boldsymbol{w}}_h^{k,s_k-1}$ are piecewise polynomial functions corresponding to $\bar{\boldsymbol{W}}_k^{s_k}$ and $\bar{\boldsymbol{W}}_k^{s_k-1}$ through isomorphism (22). Therefore, the difference criterion (37) has a nice interpretation in the framework of piecewise polynomial functions, i. e., the iterative process (31) is stopped, if the relative difference of two successive iterations measured as the $L^2(\Omega)$-norm of the corresponding functions from $\boldsymbol{S}_{h\mathsf{p}}$ is under a given tolerance.

Example 2. The efficiency and accuracy of the stopping criteria is demonstrated in Figure 2, which shows a comparison of the preconditioned residuum $Pres_k$ given by (36) and the difference criterion $Dres_k$ given by (37) for a problem with $\mathsf{dof} = 100\,600$. These figures correspond to the steady state flow simulation when the BDF-DGFE method is used for $k \to \infty$ and the computational process is stopped when

$$\eta_k \le 10^{-9}, \quad \eta_k := \frac{\frac{1}{\tau_k}\|\boldsymbol{w}_h^k - \boldsymbol{w}_h^{k-1}\|_{L^2(\Omega)}}{\frac{1}{\tau_1}\|\boldsymbol{w}_h^1 - \boldsymbol{w}_h^0\|_{L^2(\Omega)}}. \tag{39}$$

The condition (39) measures a relative decrease of the approximation of $\partial \boldsymbol{w}_h/\partial t$ in the $L^2(\Omega)$-norm. Figure 2 shows the dependencies of steady state residuum η_k (left) and number of GMRES iterations s_k (right) on $k = 1, \dots, r$. The time step was increasing exponentially for both cases according to (43). We observe that the convergence to the steady state solution is in both cases almost identical (left figure). On the other hand, the difference criterion needs significantly smaller number of GMRES iterations for larger time steps. Moreover, the total computational time was $1\,285\,s$ for preconditioned residuum and $437\,s$ for the difference criterion.

Therefore, we can deduce that the inefficiency of the block diagonal preconditioner with the large time steps is reduced if the difference stopping criterion (37) is used.

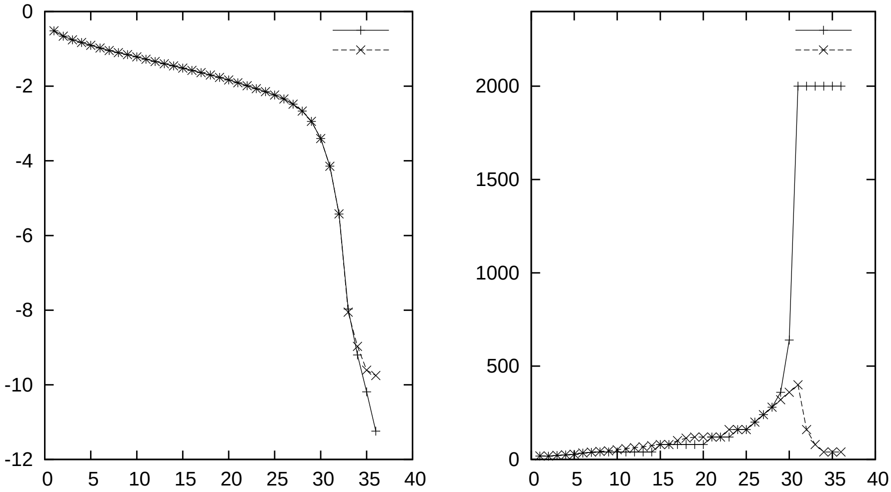

Fig. 2. Comparison of the preconditioned residuum $P\mathrm{res}_k$ (full line) and the difference criterion $D\mathrm{res}_k$ (dashed line); dependence of the steady state residuum η_k (left) and number of GMRES iterations s_k (right) on $k = 1, \dots, r$.

4.3.3. Choice of the time step

A formal strategy of the choice of the time step depends on the flow regime:

- *steady state flow*: at the beginning of a computation it is necessary to choose the time step small since we start usually from an unphysical initial condition and larger time step can cause a collapse of the computational process. On the other hand, when we are already close to the (physical) steady-state solution, the time step can be (almost) arbitrarily large since the presented BDF-DGFE method is practically unconditionally stable.
- *unsteady flow*: the time step should be kept relatively small during the whole computation in order to obtain sufficiently accurate solution with respect to time.

Usually, the adaptive choice of the time step are based on the use of two methods for ODE. From the difference of both approximate solution, we estimate of the local discretization error and propose of a new (optimal) time step, see, e. g., [18]. More precisely, let L_k denote an estimate of the local discretization error e_L^k

$$L_k \approx e_L^k := \|\boldsymbol{w}_{h,k} - \tilde{\boldsymbol{w}}_h^{k-1}(t_k)\|, \quad k = 1, \dots, r, \tag{40}$$

where $\boldsymbol{w}_{h,k}$ is the approximate solution at time level t_k and $\tilde{\boldsymbol{w}}_h^{k-1}(\cdot)$ is the exact solution of the semi-discrete problem (12) such that $\tilde{\boldsymbol{w}}_h^{k-1}(t_{k-1}) = \boldsymbol{w}_h^{k-1}$. If the ODE method has the order q then $L_k = O(\tau_k^{q+1})$. Then the time step is chosen as large as possible and satisfying the condition

$$L_k \leq \epsilon, \tag{41}$$

where $\epsilon > 0$ is a given tolerance. For more details, see [18].

However, these techniques are optimal from the point of view of the number of time steps and generally not optimal from the point of view of the computational time. It follows from the fact that the length of the time step has a great influence on the computational time of one time level, see Section 4.3.1.

In [11] we proposed the so-called *adaptive backward difference formulae* (ABDF) technique which is based on the use of two implicit multi-step methods of the same order of accuracy. Although this approach is robust for different flow regimes, the ABDF method sometimes keeps the time step not enough large in situations when the numerical solution is already close to the steady state.

Therefore, we propose the following heuristic approach. Let Λ_k be the quantity defined by

$$\Lambda_k := \max_{K\in\mathcal{T}_h} |K|^{-1} \max_{\Gamma\in\partial K} \lambda(\boldsymbol{w}_{h,k}|_\Gamma)|\Gamma|, \;\; k = 0, \ldots, r, \tag{42}$$

where $\lambda(\boldsymbol{w}_{h,k}|_\Gamma)$ is the spectral radius of matrix $\boldsymbol{P}(\boldsymbol{w}_{h,k}|_\Gamma, \boldsymbol{n}_\Gamma)$ given by (2). Then we define the size of the time step by

$$\tau_1 = \frac{1}{2\Lambda_0}, \qquad \tau_{k+1} = \frac{\eta_k^{-\delta}}{2\Lambda_k}, \quad k = 1, \ldots, r, \tag{43}$$

where η_k is given by (39) and $\delta > 0$ is a given parameter. We employ the values 3/2 or 2 usually. Obviously, $\eta_1 = 1$ and thus τ_1 and τ_2 correspond to the time steps used for the explicit time discretization with $\mathrm{CFL} = 0.5$, see [15]. The relation (43) implies that the time step τ_k is exponentially increasing for decreasing steady state residuum η_k. Moreover, from practical considerations we employ a modification of (43) in the form

$$\tau_{k+1} = \frac{\min(\eta_k^{-\delta}, 2\mathrm{CFL}_{\max})}{2\Lambda_k}, \qquad k = 1, \ldots, r, \tag{44}$$

where $\mathrm{CFL}_{\max}$ is chosen approximately $10^7 - 10^{10}$. Condition (44) prevents τ_k from assuming very large values in order to avoid some possible troubles caused by computations in the computer arithmetic. However, the presented method can realy deals with the mentioned large values of $\mathrm{CFL}_{\max}$.

Example 3. In order to demonstrate the efficiency of the proposed technique for the choice of the time step, we compare ABDF strategy [11] with the heuristic technique (HT) of the choice of the time step (42)−(44). Table 2 contains a comparison of the number of time steps and the computational time of ABDF and HT methods for two different values of ϵ and δ, respectively, for two problems with different dof. Both techniques converge to the identical steady state solution. However, HT method requires about half computational time in comparison with ABDF.

Table 2. Comparison of ABDF and HT
in terms of number of time steps and computational time in seconds.

	$\#\mathcal{T}_h = 2\,515,\ P_2,\ \mathsf{dof} = 60\,360$				$\#\mathcal{T}_h = 3\,025,\ P_3,\ \mathsf{dof} = 121\,000$			
method	ABDF		HT		ABDF		HT	
	$\epsilon = 10^{-2}$	$\epsilon = 10^{-1}$	$\delta = 3/2$	$\delta = 2$	$\epsilon = 10^{-2}$	$\epsilon = 10^{-1}$	$\delta = 3/2$	$\delta = 2$
time steps	64	54	39	17	74	63	59	20
CPU(s)	275	266	177	111	1 117	1 042	863	531

4.4. Summary of the results

In previous sections we discussed and numerically demonstrated the choice of preconditioner, stopping criterion and the choice of the time step for the solution of the sequence of linear algebraic systems (23). Based on these considerations and numerical experiments we propose to use the block diagonal preconditioner with the difference stopping criterion (37) and the heuristic choice of the time step (42)–(43). This approach significantly reduces computational time, does not require any significant increase of the computer memory and does not cause any decrease of accuracy which is demonstrated by the following example.

Example 4. In order to demonstrate the efficiency and the robustness of this approach, we compare the new proposed strategy with the explicit time discretization from [9] and our original semi-implicit discretization presented in [10] where the block diagonal preconditioner with the preconditioned stopping criterion (36) with the fixed tolerance $\omega = 10^{-4}$ and the adaptive choice of the time from [11] with $\epsilon = 10^{-2}$ were used.

In order to demonstrate the robustness of the proposed approach, we consider two different flow regimes around the profile NACA0012:

- inviscid flow with the free stream Mach number $M = 0.5$ and the angle of attack $\alpha = 2°$,
- laminar viscous flow with the free stream Mach number $M = 0.5$, the angle of attack $\alpha = 2°$ and the Reynolds number $Re = 5\,000$.

We employ adaptively refined grids having 2 515 elements for the inviscid case and 2 021 elements for the viscous case, see Figure 3. The computations were carried out using P_1 and P_3 polynomial approximations. For viscous flow, the NIPG variant of BDF-DGFE method with $C_W = 50$ in (9) was used. We compare three numerical schemes mentioned above ([9, 10] and the new one) from the point of view of the total computational time and memory requirement. The computations were stopped when the steady-state residuum η_k given by (39) was less than 10^{-9}. Nevertheless, the explicit time discretizations with P_1 polynomial approximation were stopped after 100 000 time steps when the steady state residuum was only $4 \cdot 10^{-4}$. Moreover, the computations with the explicit technique and P_3 approximation were not performed since they are much more time consuming.

All compared techniques give identical numerical results. This implies that the presented accelerations of the solution of linear algebra problem does not cause

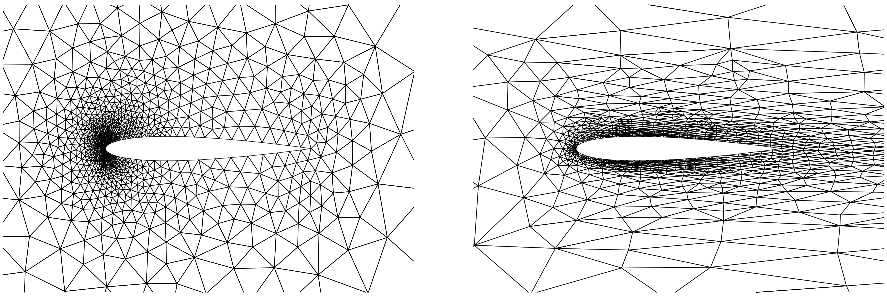

Fig 3. Adaptively refined grids used for inviscid (left) and viscous (right) flow regimes.

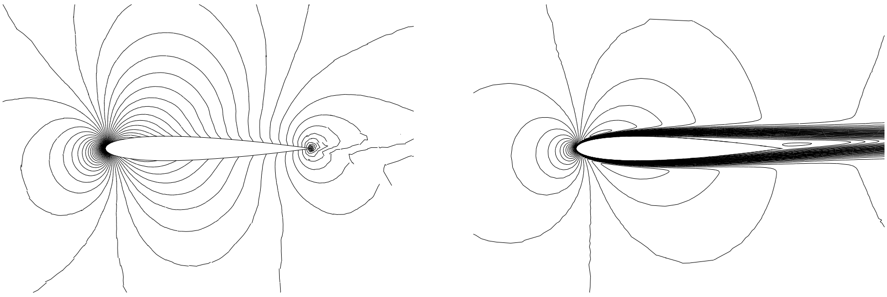

Fig. 4. Isolines of the Mach number for inviscid (left) and viscous (right) flows around the NACA0012 profile using P_3 approximation.

any loss of stability and accuracy. Table 3 shows the total computational time and the used computer memory for three tested methods for inviscid and viscous flows computed by both degrees of polynomial approximations. We observe that the semi-implicit techniques are much more efficient for the solution of steady-state flow problems than the explicit one. Moreover, the new presented semi-implicit method also significantly reduces computational time in comparison with the original approach from [10]. On the other hand, semi-implicit methods require more memory since the matrix blocks of the original matrix and preconditioner should be stored.

The significnat decrease of the computational time for the new approach (see Table 3) was achieved for the same setting of parameters $\omega_2 = 10^{-6}$ in (37) and $\delta = -3/2$ in (44 for inviscid as well as viscous flow regimes and for P_1 and P_3 polynomial approximations. Therefore, it is not necessary to tune these parameters for each case separately which indicates the robustness of the presented technique. Finally, Figure 4 shows isolines of the Mach number around the profile for both flows using P_3 approximation.

Table 3. Comparison of the explicit method [9], the semi-implicit method [10] and the new proposed method for inviscid and viscous flows, used memory and total computational time.

		P_1		P_3	
case	method	CPU time	memory	CPU time	memory
inviscid	explicit [9]	6 194 s	6 MB	—	41 MB
	implicit [10]	232 s	34 MB	2 283 s	177 MB
	new implicit	47 s	30 MB	226 s	168 MB
viscous	explicit [9]	11 680 s	5 MB	—	38 MB
	implicit [10]	362 s	25 MB	2 292 s	172 MB
	new implicit	130 s	24 MB	401 s	162 MB

5. CONCLUSION

We presented semi-implicit variant of the DGFE method for the solution of the compressible Navier–Stokes equations. We dealt with numerical solution of linear algebraic systems arising from DGFE discretization.

Based on several considerations and numerical experiments we found that the use of *block diagonal preconditioner* with the *difference stopping criterion* (37) and the *heuristic choice of the time step* (42) – (43) give the *accurate*, *robust* and *efficient* numerical scheme for the solution of the *steady-state problems*. A theoretical justification of the presented approach is the subject of the further research.

ACKNOWLEDGEMENT

This work is a part of the research project MSM 0021620839 financed by the Ministry of Education, Youth and Sport of the Czech Republic and was partly supported by the Grant No. 201/08/0012 of the Czech Science Foundation. Additionally, this work is a part of the research project ADIGMA, No. 30719 financed within the 3rd Call of the 6th European Framework Programme.

(Received March 3, 2010)

REFERENCES

[1] D. N. Arnold: An interior penalty finite element method with discontinuous elements. SIAM J. Numer. Anal. *19* (1982), 4, 742–760.

[2] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini: Unified analysis of discontinuous Galerkin methods for elliptic problems. SIAM J. Numer. Anal. *39* (2002), 5, 1749–1779.

[3] 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* (1997), 267–279.

[4] F. Bassi and S. Rebay: A high order discontinuous Galerkin method for compressible turbulent flow. In: Discontinuous Galerkin Method: Theory, Computations and Applications (B. Cockburn, G. E. Karniadakis, and C. W. Shu, eds.), (Lecture Notes in Computat. Sci. Engrg. *11*.) Springer-Verlag, Berlin 2000, pp. 113–123.

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

[6] P. G. Ciarlet: The Finite Elements Method for Elliptic Problems. North-Holland, Amsterdam – New York – Oxford 1979.

[7] 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* (1990), 545–581.

[8] C. N. Dawson, S. Sun, and M. F. Wheeler: Compatible algorithms for coupled flow and transport. Comput. Meth. Appl. Mech. Engrg. *193* (2004), 2565–2580.

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

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

[11] V. Dolejší and P. Kůs: Adaptive backward difference formula – discontinuous Galerkin finite element method for the solution of conservation laws. Internat. J. Numer. Methods Engrg. *73* (2008), 12, 1739–1766.

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

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

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

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

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

[17] M. Feistauer, V. Kučera, and J. Prokopová: Discontinuous Galerkin solution of compressible flow in time dependent domains. Math. Comput. Simulations *80* (2010), 8, 1612-1623.

[18] E. Hairer, S. P. Norsett, and G. Wanner: Solving ordinary differential equations I, Nonstiff problems. (Springer Series in Computational Mathematics No. 8.) Springer Verlag, Berlin 2000.

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

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

[21] 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* (2007), 2, 175–199.

[22] B. Rivière, M. F. Wheeler, and V. Girault: Improved energy estimates for interior penalty, constrained and discontinuous Galerkin methods for elliptic problems. I. Comput. Geosci. *3* (1999), 3-4, 337–360.

[23] D. S. Watkins: Fundamentals of Matrix Computations. (Pure and Applied Mathematics, Wiley-Interscience Series of Texts, Monographs, and Tracts.) John Wiley , New York 2002.

Vít Dolejší, Charles University in Prague, Faculty of Mathematics and Physics, Sokolovská 83, 186 75 Praha 8. Czech Republic.
e-mail: dolejsi@karlin.mff.cuni.cz