Goal-oriented space-time adaptivity in the finite element Galerkin method for the computation of nonstationary incompressible flow

Michael Besier and Rolf Rannacher*

Institute of Applied Mathematics, University of Heidelberg, Im Neuenheimer Feld 293/294, 69120 Heidelberg, Germany

SUMMARY

This paper presents a general strategy for designing adaptive space-time finite element discretizations of the nonstationary Navier-Stokes equations. The underlying framework is that of the Dual Weighted Residual (DWR) method for goal-oriented a posteriori error estimation and automatic mesh adaptation. In this approach the error in the approximation of certain quantities of physical interest, such as the drag coefficient, is estimated in terms of local residuals of the computed solution multiplied by sensitivity factors which are obtained by numerically solving an associated dual problem. In the resulting local error indicators the effects of spatial and temporal discretization are separated, which allows for the simultaneous adjustment of time step and spatial mesh size. The efficiency of the proposed method for the construction of economical meshes and the quantitative assessment of the error is illustrated by several test examples. Copyright © 2011 John Wiley & Sons, Ltd.

KEY WORDS: nonstationary incompressible flow, space-time finite elements, goal-oriented adaptivity

1. INTRODUCTION

This paper is devoted to the development of adaptive discretization techniques for the efficient numerical solution of nonstationary flow problems. The underlying mathematical model are the incompressible Navier-Stokes equations and the discretization is by the finite element Galerkin method in space and time. In contrast to stationary situations, here spatial as well as temporal discretization needs to be simultaneously controlled. This requires separate quantitative information about the respective discretization errors in order to reduce the total discretization error in a most efficient way.

Adaptive methods are widely used in the context of finite element discretizations of partial differential equations, see Verfürth [1], Eriksson et al. [2], Babuška & Strouboulis [3], and Bangerth & Rannacher [4] for an overview. In Kay et al. [5] and John & Rang [6], adaptive time-stepping methods for the incompressible Navier-Stokes equations are discussed, but using a uniform spatial discretization. In turn, in Bänsch [7] a spatially adaptive, but uniform in

*Correspondence to: Rolf Rannacher, Institute of Applied Mathematics, University of Heidelberg, Im Neuenheimer Feld 293/294, 69120 Heidelberg, Germany

Copyright © 2011 John Wiley & Sons, Ltd.
time, strategy for the nonstationary Navier-Stokes equations is developed, which is based on a posteriori error estimates in the energy-norm.

However, error estimation with respect to global norms such as the energy-norm or the $L^2$-norm is not very meaningful since in flow problems one is often mainly interested in a specific functional value of the solution, the so-called quantity of interest. Hence, the goal of the numerical simulation of a flow may be just the efficient computation of this single number. This quantity might, for instance, be the mean drag or lift coefficient of an obstacle surrounded by the fluid. In this case, the efficiency of an algorithm for numerically computing this quantity has to be measured in terms of the reduction of the discretization error in the quantity of interest rather than in global norms since the latter usually do not provide useful bounds for the error in the quantity of interest.

The results presented in this work have been developed in the PhD thesis of the first author, see [8]. They are an extension of the methodology developed in Schmich & Vexler [9] to nonstationary flow problems allowing for the simultaneous adaptation of the temporal and spatial discretization. We will derive a posteriori error estimates, which quantitatively assess the discretization error measured in the quantity of interest. In these estimates the effect of temporal and spatial discretization is separated, which allows us to set up an efficient algorithm for the simultaneous adjustment of the respective discretizations.

The key to rigorous a posteriori error estimation is a coupled variational formulation of the underlying problem. This allows the application of finite element Galerkin methods not only for the discretization in space but also for the discretization in time, together with residual-based a posteriori error estimation. Space-time Galerkin methods (referred to as General Galerkin G2) have already been applied successfully in the simulation of incompressible flows, see, for example, Mittal et al. [10], Mittal & Tezduyar [11], Behr & Tezduyar [12], or N’dri et al. [13] as well as Hoffman [14]. While the first references do not consider adaptivity, in Hoffman [14] also an adaptive algorithm is developed for nonstationary flow problems based on a posteriori error estimation, though without separating temporal and spatial discretization error.

In this paper, we consider nonstationary incompressible flows described by the incompressible Navier-Stokes equations, which read as

$$
\begin{align*}
\partial_t \mathbf{v} - \nu \Delta \mathbf{v} + \mathbf{v} \cdot \nabla \mathbf{v} + \nabla p &= \mathbf{f} \quad \text{in } I \times \Omega, \\
\nabla \cdot \mathbf{v} &= 0 \quad \text{in } I \times \Omega, \\
\mathbf{v}(0) &= \mathbf{v}^0 \quad \text{in } \Omega,
\end{align*}
$$

where $I = (0, T)$ is the time interval, $\Omega \subseteq \mathbb{R}^d$ ($d = 2, 3$) the computational domain, $\nu$ the kinematic viscosity, $\mathbf{f}$ the volume force, and $\mathbf{v}^0$ the initial value at time $t = 0$. These equations have to be supplemented by appropriate boundary conditions. For sake of simplicity, we assume no-slip Dirichlet boundary conditions, $\mathbf{v} |_{\partial \Omega} = 0$. The corresponding variational formulation reads as follows: For $\mathbf{f} \in L^2(I, H^{-1}(\Omega)^d)$ and $\mathbf{v}^0 \in L^2(\Omega)^d$ find $\mathbf{u} := (\mathbf{v}, p)^T \in X$, such that

$$
\int_I \left\{ (\partial_t \mathbf{v}, \psi) + a(\mathbf{u})(\varphi) \right\} \, dt + (\mathbf{v}(0) - \mathbf{v}^0, \psi(0)) = \int_I (\mathbf{f}, \psi) \, dt \quad \forall \varphi := (\psi, \chi)^T \in X, 
$$

(2)

with the semi-linear form

$$
a(\mathbf{u})(\varphi) := \nu (\nabla \mathbf{v}, \nabla \varphi) + (\mathbf{v} \cdot \nabla \mathbf{v}, \varphi) - (p, \nabla \cdot \varphi) + (\nabla \cdot \mathbf{v}, \chi),
$$

where $(\cdot, \cdot)$ denotes the inner product on $L^2(\Omega)$ (or $L^2(\Omega)^d$) and the space $X$ is given as

$$
X := \{ \mathbf{u} = (\mathbf{v}, p)^T, \mathbf{v} \in L^2(I, H_0^1(\Omega)^d), \partial_t \mathbf{v} \in L^2(I, H^{-1}(\Omega)^d), p \in L^2(I, L^2(\Omega)/\mathbb{R}) \}.
$$
For questions on existence and uniqueness of solutions, we refer to Temam [15].

**Remark 1.1.** In applications, we will sometimes be confronted with configurations in which Dirichlet boundary conditions for the velocity are not prescribed on the whole boundary. Instead, some part \( \Gamma_{\text{out}} \) of the boundary is an outlet. There, we apply the following artificial boundary condition:

\[
\nu \partial_n \mathbf{v} - p n = 0.
\]

This type of boundary condition implicitly normalizes the pressure such that it is already uniquely determined without the usual mean value constraint. Hence, the spaces in which the solutions are sought have to be modified to

\[
\mathbf{v} \in \left\{ \mathbf{v} \in H^1(\Omega)^d, \mathbf{v}|_{\partial \Omega \setminus \Gamma_{\text{out}}} = 0 \right\}, \quad p \in L^2(\Omega).
\]

For more information on this boundary condition (so-called “do nothing” condition) as well as results concerning existence and uniqueness of solutions, we refer to Heywood et al. [16].

Furthermore, we consider a functional \( J : X \to \mathbb{R} \) representing the quantity of physical interest. This functional is given as a sum

\[
J(u) = \int_0^T J_1(u(t)) \, dt + J_2(u(T)),
\]

where \( J_1 \) or \( J_2 \) may be zero. Let \( u_{kh} = (v_{kh}, p_{kh})^T \) be a solution of the discretized version of problem (2). Then, we aim at the a posteriori error estimation with respect to \( J \) of the following type:

\[
J(u) - J(u_{kh}) \approx \eta_k + \eta_h,
\]

where \( \eta_k \) measures the error due to the discretization in time and \( \eta_h \) the error due to the discretization in space.

The outline of this paper is as follows: In Section 2, we describe the space-time finite element discretization of problem (2). Section 3 is devoted to the derivation of a posteriori error estimates for the discretization error with respect to the quantity of interest \( J(u) \). This analysis is developed on a rather abstract level, while the technical details for the Navier-Stokes equations are given in the Appendix. These error estimates separately assess the error due to the discretization in time and in space and are obtained by using the solution of a (linear) dual problem. Further, we describe the numerical realization of the derived error estimates and an adaptive algorithm for successive improvement of the accuracy. In Section 4 numerical results are presented illustrating the performance of the adaptive method.

### 2. DISCRETIZATION

In this section, we describe the discretization of the weak formulation of the incompressible Navier-Stokes equations (2) by the Rothe method. The discretization in space as well as in time will be done by means of finite element Galerkin methods. At first, we present the semi-discretization in time by discontinuous Galerkin (dG) methods. Then, we introduce the discretization in space of the arising semi-discrete problems. This uses continuous Galerkin (cG) methods. For technical reasons, we use piecewise polynomial functions of the same degree for the velocity and the pressure component, which do not satisfy the Babuška-Brezzi stability condition. Therefore, we have to apply suitable pressure stabilization.
2.1. Discretization in time

To introduce the semi-discretization in time, we partition the time interval $\bar{I} = [0, T]$, with subintervals $I_m := (t_{m-1}, t_m]$ of length $k_m := t_m - t_{m-1}$, using time points

$$0 = t_0 < t_1 < \cdots < t_m < \cdots < t_M = T.$$ 

The discretization parameter $k$ is given as a piecewise constant function by $k|_{I_m} := k_m$ for $m = 1, \ldots, M$. On the subintervals $I_m$, we define the following semi-discrete spaces $X_k^r$ for $r \in \mathbb{N}_0$:

$$X_k^r := \left\{ \mathbf{u}_k = (\mathbf{v}_k, p_k)^T, \frac{\partial \mathbf{v}_k}{\partial t}|_{I_m} \in L^2(\Omega)^d, \mathbf{v}_k|_{I_m} \in \mathcal{P}_r(I_m, H^1_0(\Omega)^d), \right.$$ 

$$p_k|_{I_m} \in \mathcal{P}_r(I_m, L^2(\Omega)/\mathbb{R}), \quad m = 1, \ldots, M \} \subseteq L^2(I, H^1_0(\Omega)^d \times L^2(\Omega)/\mathbb{R})$$

where $\mathcal{P}_r(I_m, Y)$ denotes the space of polynomials up to degree $r$ on $I_m$ with values in $Y$. To account for the possible discontinuity of a function $u_k$ at time points $t_m$, we introduce the notation

$$u_{k,m}^+ := \lim_{\varepsilon \downarrow 0} u_k(t_m + \varepsilon), \quad u_{k,m}^- := \lim_{\varepsilon \downarrow 0} u_k(t_m - \varepsilon), \quad [u_k]_m := u_{k,m}^+ - u_{k,m}^-,$$

i.e., $u_{k,m}^+$ and $u_{k,m}^-$ are the limits “from above” and “from below” at time $t_m$, respectively, while $[u_k]_m$ is the corresponding “jump” of $u_k$ at $t_m$ (see Figure 1).

Then, the $dG(r)$ semi-discretization of the incompressible Navier-Stokes equations (2) reads: Find $\mathbf{u}_k = (\mathbf{v}_k, p_k)^T \in X_k^r$ such that

$$\sum_{m=0}^{M-1} \int_{I_m} \left\{ (\partial_t \mathbf{v}_k, \psi) + a(\mathbf{u}_k)(\varphi) \right\} \, \mathrm{d}t + \sum_{m=0}^{M-1} \left[ (\mathbf{v}_{k,m}^+, \psi_m^+), (\mathbf{v}_{k,0}, \psi_0^-) \right]$$

$$= \int_{I} (f, \psi) \, \mathrm{d}t + (\mathbf{v}_0^+, \psi_0^-) \quad \forall \varphi = (\psi, \chi)^T \in X_k^r.$$ 

(5)

**Remark 2.1.** Due to the discontinuity of the test functions, the $dG(r)$ discretization decouples into a time stepping scheme. For example, the $dG(0)$ discretization is a variant of the backward Euler method, while the $dG(1)$ discretization, after applying quadrature rules to the temporal integrals, corresponds to an implicit Range-Kutta method.
2.2. Discretization in space

Next, we describe the discretization in space of the semi-discrete problems obtained in the previous subsection. To this end, we use two- or three-dimensional shape-regular meshes, see, e.g., Ciarlet [17]. A mesh consists of (closed) quadrilateral or hexahedral cells $K$ with non-overlapping interiors, which form a covering of the computational region $\Omega \subseteq \mathbb{R}^d$. The corresponding mesh is denoted by $T_h = \{K\}$, where the discretization parameter $h$ is defined as a cellwise constant function by setting $h|_K = h_K$ with the diameter $h_K$ of the cell $K$.

**Remark 2.2.** In order to ease mesh refinement, we allow cells to have nodes which lie on midpoints of faces or edges of neighboring cells. But at most one such “hanging node” is permitted on each face or edge. There are no degrees of freedom associated to these irregular nodes as there the value of a finite element function is determined by interpolation of neighboring nodal values, see Carey & Oden [18] for more details.

On the mesh $T_h$, we construct a conforming finite element space $V_{h}^s \subseteq H^1(\Omega)$ in a standard way:

$$V_{h}^s := \{v \in C(\Omega), v|_K \in Q_s(K) \text{ for } K \in T_h\}.$$  

We use isoparametric elements, i.e., $Q_s(K)$ consists of shape functions obtained via $\tilde{Q}_s(\tilde{K})^d$ transformations of polynomials in $\tilde{Q}_s(\tilde{K})$ defined on the reference cell $\tilde{K} = (0,1)^d$,

$$\tilde{Q}_s(\tilde{K}) = \text{span} \left\{ \prod_{j=1}^{d} x_j^{\alpha_j}, \alpha_j \in \{0, \ldots, s\} \right\}.$$  

In addition, we require that the mesh is organized in a patchwise manner. That is, $T_h$ is obtained by uniform refinement of a coarser mesh $T_{2h}$, such that we can always combine four ($d = 2$) or eight ($d = 3$) adjacent cells of $T_h$ to obtain one cell of $T_{2h}$. Such macro-cells are called patches (see Figure 2).

To obtain the formulation of the fully discrete problem, we allow dynamic mesh change in time, but the local time steps $k_m$ are kept constant in space. To this end, with each time point $t_m$, we associate a mesh $T_{h,m}$ and corresponding (spatial) finite element spaces $V_{h}^{s,m}$ and $V_{h}^{p,m}$. Then, we define the following space-time finite element space:

$$X_{kh}^{r,s} := \left\{ u_{kh} = (v_{kh}, p_{kh})^T, \, v_{kh}(0) \in (H^0_0)^d, \, v_{kh}|_{t_m} \in P_r(I_m, (H^0_0)^d), \, p_{kh}|_{t_m} \in P_r(I_m, L^2_0)^d, \, m = 1, \ldots, M \right\} \subseteq L^2(I, H^1_0(\Omega)^d \times L^2(\Omega)/\mathbb{R}),$$

where $I_m$ denotes the time interval $I_m = [t_{m-1}, t_m)$. The space-time finite element space $X_{kh}^{r,s}$ is defined in such a way that it contains all finite element solutions that are smooth in time and space up to a certain level of regularity $r$.

**Figure 2.** Two-dimensional mesh $T_h$ (with hanging nodes) organized in a patchwise manner with corresponding coarser mesh $T_{2h}$.
where \( H_{h}^{m} : = V_{h}^{s,m} \cap H_{0}^{1}(\Omega) \), \( L_{h}^{m} : = V_{h}^{s,p,m} \cap L^{2}(\Omega)/\mathbb{R} \).

Due to the conformity of \( H_{h}^{m} \) and \( L_{h}^{m} \), we have \( X_{h}^{r,s} \subseteq X_{k}^{r} \). Then, the fully discrete \( \text{cG}(s) \text{dG}(r) \) formulation of problem (2) reads: Find \( u_{kh} = (v_{kh}, p_{kh})^{T} \in X_{h}^{r,s} \), such that
\[
\sum_{m=1}^{M} \int_{I_{m}} \left\{ (\partial_{t} v_{kh}, \psi) + a(u_{kh})(\varphi) \right\} \, dt + \sum_{m=0}^{M-1} \left( [v_{kh}]_{m}, \psi_{m}^{+} \right) + \left( v_{kh,0}, \psi_{0}^{-} \right) = \int_{I} (f, \psi) \, dt + (v_{0}^{0}, \psi_{0}^{-}) \quad \forall \varphi = (\psi, \chi)^{T} \in X_{h}^{r,s}.
\]

**Remark 2.3.** The notation \( \text{cG}(s) \text{dG}(r) \) for a space-time finite element Galerkin discretization with continuous piecewise polynomials of degree \( s \) in space and discontinuous piecewise polynomials of degree \( r \) in time, is adopted from Eriksson et al. [19].

### 2.3. Stabilization

The fully discrete formulation described above does not lead to a stable approximation of problem (2) unless the spatial finite element spaces \( H_{h}^{m} \) and \( L_{h}^{m} \) fulfill the Babuška-Brezzi inf-sup-stability condition. Especially the cases of equal-order trial spaces, i.e., \( s_{v} = s_{p} = s \), are favorable from the implementation point of view, do not fulfill this condition. Therefore, we either have to use special stable finite elements like the Taylor-Hood element (see Hood & Taylor [20]) or in general add stabilization terms. Here, for ease of implementation, we use equal-order trial spaces and apply the local projection stabilization (LPS), see, e.g., Becker & Braack [21, 22].

To give a precise definition of the modified fully discrete formulations, we introduce a spatial interpolation operator \( I_{h} : V_{h}^{s,m} \rightarrow \tilde{V}_{h}^{s,m} \) into a subspace \( \tilde{V}_{h}^{s,m} \subseteq V_{h}^{s,m} \), which is given as
\[
\tilde{V}_{h}^{s,m} : = \begin{cases} V_{2h}^{1,m} & \text{for } s = 1, \\ V_{1h}^{1,m} & \text{for } s = 2. \end{cases}
\]

The interpolation onto the mesh \( \mathcal{T}_{2h} \) in the case \( s = 1 \) is easily computable if the mesh possesses the patch structure introduced above. Using the interpolation operator \( I_{h} \), we define the filtering operator \( \pi : V_{h}^{s,m} \rightarrow V_{h}^{s,m} \) by
\[
\pi := \text{id} - I_{h}.
\]

The filtering operator \( \pi : (V_{h}^{s,m})^{d} \rightarrow (V_{h}^{s,m})^{d} \) is defined analogously componentwise. Let us further extend these operators in time by setting
\[
(\pi p_{kh})(t) : = \pi p_{kh}(t), \quad (\pi v_{kh})(t) : = \pi v_{kh}(t).
\]

This allows us to state the following modified fully discrete formulation of problem (2): Find \( u_{kh} = (v_{kh}, p_{kh})^{T} \in X_{h}^{r,s} \), such that
\[
\sum_{m=1}^{M} \int_{I_{m}} \left\{ (\partial_{t} v_{kh}, \psi) + a(u_{kh})(\varphi) + s_{m}^{0}(u_{kh})(\varphi) \right\} \, dt + \sum_{m=0}^{M-1} \left( [v_{kh}]_{m}, \psi_{m}^{+} \right) + \left( v_{kh,0}, \psi_{0}^{-} \right) = \int_{I} (f, \psi) \, dt + (v_{0}^{0}, \psi_{0}^{-}) \quad \forall \varphi = (\psi, \chi)^{T} \in X_{h}^{r,s}.
\]
Here, the additional terms are given by

\[ s_h^W(u)(\varphi) := \sum_{K \in T_h^m} \left\{ (\nabla p, \alpha_{K,m} \nabla \pi \chi)_K + v \cdot \nabla \pi v, q_{K,m} v \cdot \nabla \psi)_K \right\}, \]

with \( u = (v, p)^T \) and \( \varphi = (\psi, \chi)^T \). The cellwise stabilization parameters \( \alpha_{K,m} \) and \( \delta_{K,m} \) are defined as

\[ \alpha_{K,m} = \alpha_0 \frac{h_K^2}{6\nu + h_K \|v_{kh}\|_K}, \quad \delta_{K,m} = \delta_0 \frac{h_K^2}{6\nu + h_K \|v_{kh}\|_K + h_K/k_m}, \]

with some constants \( \alpha_0 \) and \( \delta_0 \). For details on the choice of these parameters, we refer to Franca & Frey [23] or Braack et al. [24]. In our computations, we chose \( \alpha_0 = \delta_0 = 0.3 \).

3. A POSTERIORI ERROR ESTIMATION AND MESH ADAPTATION

This section is devoted to the development of an a posteriori error estimator which measures the discretization error in the output value \( J(u) \). The error estimator developed here is an extension to the Navier-Stokes equations of concepts developed in Schmich & Vexler [9] for scalar parabolic equations. It separates the total discretization error into contributions due to the discretization in time and in space. The reliable quantitative error estimation is a key ingredient in setting up an adaptive algorithm by which the temporal and spatial discretization errors are balanced and simultaneously decreased. Here, we only state these results in a rather abstract form using a general framework of optimization theory. The technical details for the Navier-Stokes equations are given in the Appendix.

3.1. A posteriori error estimation

As mentioned in Section 1, we assume the error functional \( J(\cdot) \) to be given in the form

\[ J(u) = \int_0^T J_1(u(t)) \, dt + J_2(u(T)), \]

where \( J_1(\cdot) \) or \( J_2(\cdot) \) may be zero. We introduce the Lagrangian functionals \( \tilde{\mathcal{L}} : X \times X \to \mathbb{R} \), \( \tilde{\mathcal{L}}_h : X_h^r \times X_h^r \to \mathbb{R} \), and \( \tilde{\mathcal{L}}_h : X_h^{r,s} \times X_h^{r,s} \to \mathbb{R} \) by

\[ \tilde{\mathcal{L}}(u, z) := J(u) + \int \left\{ (f - \partial_t v, w) - a(u)(z) \right\} \, dt - (v(0) - v_0, w(0)), \]

\[ \tilde{\mathcal{L}}_h(u_k, z_k) := J(u_k) + \sum_{m=1}^{M-1} \int \left\{ (f - \partial_t v_k, w_k) - a(u_k)(z_k) \right\} \, dt \]

\[ - \sum_{m=0}^{M-1} (|v_k|_m, w_{k,m}^+) - (v_{k,0} - v_0, w_{k,0}^+), \]

\[ \tilde{\mathcal{L}}_h(u_{kh}, z_{kh}) := \tilde{\mathcal{L}}(u_{kh}, z_{kh}) - S_h(u_{kh}, z_{kh}), \]

with

\[ S_h(u_{kh}, z_{kh}) := \sum_{m=1}^{M} \int_0^t s_h^W(u_{kh})(z_{kh}) \, dt. \]
Here, the Lagrange multipliers $z = (w, q)^T$, $z_h = (w_h, q_h)^T$, and $z_{kh} = (w_{kh}, q_{kh})^T$ are called “dual variables” in contrast to the primal variables $u = (v, p)^T$, $u_h = (v_h, p_h)^T$, and $u_{kh} = (v_{kh}, p_{kh})^T$. Then, the variational equations satisfied by the continuous solution $u$, the semi-discrete solution $u_h$ of the $dG(r)$ discretization in time, and the fully discrete solution $u_{kh}$ of the $cG(s)dG(r)$ discretization can be identified with the equations

\[
L'_z(u, z)(\varphi) = 0 \quad \forall \varphi \in X,
\]

\[
\tilde{L}_z(u_h, z_h)(\varphi) = 0 \quad \forall \varphi \in X_k^s,
\]

\[
\tilde{L}_{h,z}(u_{kh}, z_{kh})(\varphi) = 0 \quad \forall \varphi \in X_{kh}^{rs},
\]

which involve the directional derivatives of the Lagrangians with respect to their second argument. The latter are defined as so-called Gâteaux derivatives, e.g.,

\[
L'_z(u, z)(\varphi) := \lim_{\tau \neq 0, \tau \to 0} \tau^{-1}\{L(u, z + \tau \varphi) - L(u, z)\}, \quad \varphi \in X.
\]

Correspondingly, the continuous, semi-discrete, and fully discrete dual solutions $z \in X$, $z_h \in X_k^s$, and $z_{kh} \in X_{kh}^{rs}$ are determined by the variational equations

\[
L'_u(u, z)(\varphi) = 0 \quad \forall \varphi \in X,
\]

\[
\tilde{L}_u(u_h, z_h)(\varphi) = 0 \quad \forall \varphi \in X_k^s,
\]

\[
\tilde{L}_{h,u}(u_{kh}, z_{kh})(\varphi) = 0 \quad \forall \varphi \in X_{kh}^{rs},
\]

this time involving the directional derivatives of the Lagrangians with respect to their first argument. Since we want to separate the influences of the temporal and spatial discretization, we split the total discretization error like

\[
J(u) - J(u_{kh}) = J(u) - J(u_h) + J(u_h) - J(u_{kh}).
\]

For the two error terms on the right-hand side, we have the following representations, which are stated in Theorem 5.2 of the Appendix:

\[
J(u) - J(u_h) = \frac{1}{2} \tilde{L}_z(u_h, z_h)(u - \tilde{u}_h, z - \tilde{z}_h) + R_h,
\]

\[
J(u_h) - J(u_{kh}) = \frac{1}{2} \tilde{L}_u(u_{kh}, z_{kh})(u_h - \tilde{u}_{kh}, z_h - \tilde{z}_{kh})
\]

\[+ \frac{1}{2} S_h(u_{kh}, z_{kh})(\tilde{u}_{kh} - u_{kh}, \tilde{z}_{kh} - z_{kh}) + S_h(u_{kh}, z_{kh}) + R_h.
\]

Here, $(\tilde{u}_h, \tilde{z}_h)^T \in X_k^s \times X_k^s$ and $(\tilde{u}_{kh}, \tilde{z}_{kh})^T \in X_{kh}^{rs} \times X_{kh}^{rs}$ can be chosen arbitrarily and the remainder terms $R_h$ and $R_{kh}$ are cubic in the errors $u - u_h$, $z - z_h$, and $u_h - u_{kh}$, $z_h - z_{kh}$, respectively. Now, introducing the primal and dual residuals

\[
\rho(u)(\varphi) := L'_z(u, z)(\varphi), \quad \rho^*(u, z)(\varphi) := \tilde{L}_u(u, z)(\varphi),
\]

these representations may be rewritten as

\[
J(u) - J(u_h) \approx \frac{1}{2} \left\{ \rho(u_h)(z - \tilde{z}_h) + \rho^*(u_h, z_h)(u - \tilde{u}_h) \right\},
\]

\[
J(u_h) - J(u_{kh}) \approx \frac{1}{2} \left\{ \rho(u_{kh})(z_h - \tilde{z}_{kh}) + \rho^*(u_{kh}, z_{kh})(u_h - \tilde{u}_{kh}) \right\},
\]

where we have neglected the remainder terms $R_h$ and $R_{kh}$ as well as the additional terms due to stabilization. The latter can be assumed to be small compared to the other residual terms because they contain small stabilization parameters. This is confirmed by our numerical results presented in Section 4.
3.2. Practical realization

Next, we give details on the numerical realization of the a posteriori error estimators developed in the previous section. The error estimators involve the continuous, semi-discrete, and fully discrete dual solutions \( z \in X, \ z_k \in X^*_k, \) and \( z_{kh} \in X^{r,s}_{kh}, \) respectively. These are given as solutions of the variational equations (10). We want to derive the precise form of the directional derivatives in (10). The continuous dual solution \( z = (w,q)^T \in X \) is the solution of

\[
\int_I \left\{ (\psi, -\partial_t w) + a'(u)(\varphi, z) \right\} \mathrm{d}t + (\psi(T), w(T)) = \int_I J_1'(u)(\varphi) \mathrm{d}t + J_2'(u(T))(\varphi(T)) \quad \forall \varphi = (\psi, \chi)^T \in X,
\]

where we have integrated by parts, which is allowed for functions in \( X, \) see, for instance, Wloka [26]. In the present case of the Navier-Stokes equations the derivative \( a'(u)(\varphi, z) \) of the “energy form” \( a(u)(z) \) has the explicit form

\[
a'(u)(\varphi, z) = \nu(\nabla \psi, \nabla w) + (\psi \cdot \nabla v, w) + (\psi \cdot \nabla v, w) - (\psi, \nabla \cdot w) + (\nabla \cdot \psi, q),
\]

where \( u = (v,p)^T, \ z = (w,q)^T, \ \varphi = (\psi,\chi)^T. \) The semi-discrete dual solution \( z_k = (w_k,q_k)^T \in X^*_k \) and the fully discrete dual solution \( z_{kh} = (w_{kh},q_{kh})^T \in X^{r,s}_{kh} \) satisfy the following analogous equations:

\[
\sum_{m=1}^M \int_{I_m} \left\{ (\psi, -\partial_t w_k) + a'(u_k)(\varphi, z_k) \right\} \mathrm{d}t - \sum_{m=0}^{M-1} (\psi_m, [w_k]_m) + (\psi_M, w_{k,M}) = \int_I J_1'(u_k)(\varphi) \mathrm{d}t + J_2'(u_{k,M})(\varphi_M) \quad \forall \varphi = (\psi,\chi)^T \in X^*_k,
\]

and

\[
\sum_{m=1}^M \int_{I_m} \left\{ (\psi, -\partial_t w_{kh}) + a'(u_{kh})(\varphi, z_{kh}) + s'_h(u_{kh})(\varphi, z_{kh}) \right\} \mathrm{d}t - \sum_{m=0}^{M-1} (\psi_m, [w_{kh}]_m) + (\psi_M, w_{kh,M}) = \int_I J_1'(u_{kh})(\varphi) \mathrm{d}t + J_2'(u_{kh,M})(\varphi_M) \quad \forall \varphi = (\psi,\chi)^T \in X^{r,s}_{kh}.
\]

**Remark 3.1.** We note that for solving the dual problems (18), the primal solution \( u_{kh} \) is needed on the whole time interval \( I \) due to the nonlinearity of the primal problem. A common way to deal with this difficulty is to apply checkpointing techniques which reduce the required amount of memory because the primal solution is only stored on so-called checkpoints. The drawback is that we have to solve the (nonlinear) primal problem more often to recover the primal solution between two checkpoints. More information on “checkpointing” can be found in Griewank [27], Berggren et al. [28] or Walther & Griewank [29]. However, since in the last years the capacity of main memory and hard disk drives has been growing rapidly, we propose to store the primal solution over the whole time interval. For two-dimensional simulations this can often be done by only using the main memory, while in three space dimensions, we suggest storing the data on hard disk. Even though the access of reading and writing from and to hard disk is much slower than the access to main memory, this can be assumed to be still much faster than solving several time steps of the nonlinear primal problem more than once. For a discussion of this topic and numerical experiments, we also refer to Meidner [30].
Let us now consider the numerical evaluation of the error estimator developed in the previous section for the \( cG(s)\)dG(0) and \( cG(s)\)dG(1) discretization with \( s \in \{1,2\} \). Since the quantities \( \tilde{u}_k, \tilde{z}_k, \tilde{u}_{kh}, \) and \( \tilde{z}_{kh} \) can be chosen arbitrarily in the corresponding spaces, the so-called weights, i.e., \( u - \tilde{u}_k, z - \tilde{z}_k, \) and so on, are essentially interpolation errors. We approximate these interpolation errors by higher order reconstruction from the discrete solutions. This approach relies on the “super-closeness” of the derivatives of these higher order interpolations to those of the continuous solution (see Becker & Rannacher [25] for more details on this topic and alternative approaches).

We introduce the following linear operators for approximating the weights in the error estimator:

\[
\begin{align*}
v - \tilde{v}_k & \approx \Pi_k^{(u)} v_k, & v_k - \tilde{v}_{kh} & \approx \Pi_h^{(u)} v_{kh}, \\
p - \tilde{p}_k & \approx \Pi_k^{(p)} p_k, & p_k - \tilde{p}_{kh} & \approx \Pi_h^{(p)} p_{kh}, \\
w - \tilde{w}_k & \approx \Pi_k^{(w)} w_k, & w_k - \tilde{w}_{kh} & \approx \Pi_h^{(w)} w_{kh}, \\
q - \tilde{q}_k & \approx \Pi_k^{(q)} q_k, & q_k - \tilde{q}_{kh} & \approx \Pi_h^{(q)} q_{kh}.
\end{align*}
\]

The operators \( \Pi_k^{(u)}, \Pi_k^{(p)} \) as well as \( \Pi_h^{(u)}, \Pi_h^{(p)} \) are chosen for the various discretizations as follows:

- \( cG(s)\)dG(0) scheme:
  \[
  \begin{align*}
  \Pi_k^{(u)} & := I_k^{(1)} - \text{id}, & \Pi_h^{(u)} & := I_{2h}^{(2s)} - \text{id}, \\
  \Pi_k^{(p)} & := I_k^{(1)} - \text{id}, & \Pi_h^{(p)} & := I_{2h}^{(2s)} - \text{id},
  \end{align*}
  \]
  where \( I_k^{(1)} \) is given as in Figure 3(a) and \( I_k^{(1)} \) acts componentwise like \( I_k^{(1)} \).

- \( cG(s)\)dG(1) scheme:
  \[
  \begin{align*}
  \Pi_k^{(u)} & := I_{2h}^{(2)} - \text{id}, & \Pi_h^{(u)} & := I_{2h}^{(2s)} - \text{id}, \\
  \Pi_k^{(p)} & := I_{2h}^{(2)} - \text{id}, & \Pi_h^{(p)} & := I_{2h}^{(2s)} - \text{id},
  \end{align*}
  \]
  where \( I_{2h}^{(2)} \) is given as in Figure 3(b) and \( I_{2h}^{(2)} \) acts componentwise like \( I_{2h}^{(2)} \).

The spatial interpolation operators \( I_{2h}^{(2)} : V_1^2 \rightarrow V_{2h}^2 \) (for \( s = 1 \)) into the space of bi- or tri-quadratic trial functions and \( I_{2h}^{(4)} : V_2^2 \rightarrow V_{2h}^2 \) (for \( s = 2 \)) into the space of bi- or tri-quartic trial functions, respectively, can easily be computed if the underlying mesh possesses a patch structure (see Figure 2). These spatial interpolation operators are extended into time by setting

\[
(I_{2h}^{(2s)} u_{kh})(t) := I_{2h}^{(2s)} (u_{kh}(t)).
\]

The last step in making the derived a posteriori error estimators computable is to replace all unknown solutions appearing either in the weights or in the residuals by the computed fully discrete solutions, i.e., we approximate

\[
\rho(u_k)(z - \tilde{z}_k) \approx \rho(u_{kh})(\Pi_k z_{kh}),
\]

and

\[
\rho^*(u_k, z_k)(u - \tilde{u}_k) \approx \rho^*(u_{kh}, z_{kh})(\Pi_k u_{kh}),
\]

where \( \Pi_k z_{kh} := (\Pi_k^{(u)} w_{kh}, \Pi_k^{(p)} q_{kh})^T \) and \( \Pi_k u_{kh} := (\Pi_k^{(u)} v_{kh}, \Pi_k^{(p)} p_{kh})^T \).
Remark 3.2. The above approximation is considered as justified in the weights while it may seem critical in the residuals, i.e., for the point of linearization. One could think of replacing the unknown solutions also by higher order interpolations as we do in the weights. However, in our numerical examples, we see that this additional effort is not necessary to obtain quantitatively good results. This observation is also substantiated by the fact that the approximation of the linearization point introduces an additional error which usually is of higher order, which can be seen as follows: The introduced error can be expressed as

\[ \tilde{E}'(\zeta_k)(\zeta - \tilde{\zeta}_k) - \tilde{E}'(\zeta_{kh})(\zeta - \tilde{\zeta}_k) = \int_0^1 \tilde{E}''(\zeta_{kh} + s(\zeta_k - \zeta_{kh}))(\zeta_k - \zeta_{kh}, \zeta - \tilde{\zeta}_k) \, ds, \]

where \( \zeta = (u, z)^T, \zeta_k = (u_k, z_k)^T, \) and \( \zeta_{kh} = (u_{kh}, z_{kh})^T. \) By choosing an appropriate interpolant for \( \tilde{\zeta}_k, \) this identity shows that the discussed approximation introduces an error of the order \( O(h^2k) \), whereas the total discretization error usually is not better than \( O(h^2 + k) \) in the case of a \( cG(1)dG(0) \) discretization. For more details on this issue, we refer to Meidner [30] or Schmich & Vexler [9].

Proceeding as proposed, we obtain the following global a posteriori error estimator

\[ J(u) - J(u_{kh}) \approx \eta_k + \eta_h, \]

(19)
where
\[
\eta_k := \frac{1}{2} \left\{ \rho(u_{kh})(\Pi_k z_{kh}) + \rho^* (u_{kh}, z_{kh})(\Pi_k u_{kh}) \right\},
\]
\[
\eta_h := \frac{1}{2} \left\{ \rho(u_{kh})(\Pi_h z_{kh}) + \rho^* (u_{kh}, z_{kh})(\Pi_h u_{kh}) \right\}.
\]

The concrete form of these error estimators will be developed in the Appendix.

The a posteriori error estimators developed above serve two purposes: first the quantitative assessment of the discretization error and secondly the adaptive adjustment of the underlying discretizations in order to efficiently improve the accuracy. For the second aim, the information of the error estimators has to be localized to cellwise or nodewise contributions. These quantities are then called local error indicators. To this end, we split the total error estimators into their contributions on each subinterval \( I_m \),
\[
\eta_k = \sum_{m=1}^{M} \eta_k^m, \quad \eta_h = \sum_{m=0}^{M} \eta_h^m,
\]
where the intervalwise error estimators are defined analogously to the global error estimators,
\[
\eta_k^m := \frac{1}{2} \left\{ \rho_m(u_{kh})(\Pi_k z_{kh}) + \rho^*_m (u_{kh}, z_{kh})(\Pi_k u_{kh}) \right\},
\]
\[
\eta_h^m := \frac{1}{2} \left\{ \rho_m(u_{kh})(\Pi_h z_{kh}) + \rho^*_m (u_{kh}, z_{kh})(\Pi_h u_{kh}) \right\}.
\]

These terms only involve those parts \( \rho_m \) and \( \rho^*_m \) of the global residuals \( \rho \) and \( \rho^* \) belonging to the subinterval \( I_m \) or to the initial time \( t = 0 \) for \( m = 0 \).

While the hereby obtained local contributions \( \eta_k^m \) for the temporal discretization error can directly be used for an adaptive refinement of the temporal discretization, the spatial contributions have to be localized further,
\[
\eta_h^m = \sum_{K \in T_m} \eta_{h,K}^m, \quad m = 0, \ldots, M.
\]
However, a simple splitting into cellwise contributions leads to a large overestimation of the actual error due to the oscillatory behavior of the residuals (see Carstensen & Verfürth [31]). The commonly used way to overcome this difficulty is to apply cellwise integration by parts, see, e.g., Becker & Rannacher [32, 25]. The resulting local error indicators involve strong equation residuals of the discrete solution as well as certain jump residuals across cell interfaces. The detailed description of this approach for the Navier-Stokes equations is given in the Appendix.

3.3. Mesh adaptation

For setting up an efficient adaptive algorithm, it is essential that the temporal error estimator \( \eta_k \) is independent of the refinement of the spatial discretization and vice versa. We will see in Section 4 that this is (almost) the case for both estimators, \( \eta_k \) and \( \eta_h \). However, the local contributions \( \eta_h^m \) depend linearly on the local size \( k_m \) of the time intervals \( I_m \) because
\[
\eta_h = \sum_{m=0}^{M} \eta_h^m.
\]
It is important to get rid of this dependence since otherwise the spatial error indicators would decrease, for instance, while keeping the spatial discretization fixed and only refining
the temporal discretization. Therefore, we introduce spatial error indicators \( \hat{\eta}_{m}^{h,K} \), which by rescaling are independent of \( k_m \) and hence are usable in a simultaneous mesh adaptation algorithm as presented below,

\[
\hat{\eta}_{h,K}^{m} := \frac{T}{k_m} \hat{\eta}_{h,K}^{m}, \quad K \in T_{h}^{m}, \quad m = 0, \ldots, M.
\]

This scaling has the following property: If the rescaled spatial error indicators \( \hat{\eta}_{h}^{m} \) fulfill

\[
\hat{\eta}_{h}^{m} = \sum_{K \in T_{h}^{m}} \hat{\eta}_{h,K}^{m} < \text{TOL},
\]

then, for the whole spatial error estimator, we obtain

\[
\eta_{h} = \sum_{m=0}^{M} \eta_{h}^{m} = \sum_{m=0}^{M} \frac{k_m}{T} \hat{\eta}_{h}^{m} < \frac{\text{TOL}}{T} \sum_{m=0}^{M} k_m = \text{TOL}.
\]

**Remark 3.3.** If the cells to be refined within an adaptive mesh refinement procedure are not chosen by a tolerance-based selection criterion, the simpler rescaling

\[
\hat{\eta}_{h,K}^{m} := \frac{1}{k_m} \eta_{h,K}^{m}
\]

is sufficient.

Now, we are provided with two sets of error indicators, which will be used within the adaptive algorithm presented below for an efficient automatic adaptation of the temporal and spatial discretizations. These sets are given by

\[
\Sigma_{k} := \{ \eta_{h}^{m}, \quad m = 1, \ldots, M \}, \quad \Sigma_{h} := \{ \hat{\eta}_{h,K}^{m}, \quad K \in T_{h}^{m}, \quad m = 0, \ldots, M \}.
\]

**Remark 3.4.** We note that for efficiency reasons it is necessary to treat the cellwise error indicators of all spatial meshes simultaneously rather than for each mesh separately. If we used \( M+1 \) different sets of cellwise error indicators

\[
\Sigma_{h}^{m} := \{ \hat{\eta}_{h,K}^{m}, \quad K \in T_{h}^{m} \}, \quad m = 0, \ldots, M,
\]

for deciding, which cells should be refined, we would probably obtain a rather inefficient spatial discretization. This becomes clear if we assume, for example, that the error indicators on one subinterval are much smaller than those on another subinterval. Using, for instance, a “fixed fraction strategy” for selecting the cells to be refined leads to cells that are marked to be refined although the corresponding error indicators are smaller than the error indicators of cells in other meshes which might not be refined. In turn, we also observe inefficiency when marking cells for coarsening. Even if their error indicators might be small compared to other cells in the same mesh, their contribution to the spatial discretization error still might be large compared to cells in other meshes. However, applying a “fixed fraction strategy” to the full set of error indicators \( \Sigma_{h} \) does not produce such inefficient meshes because the error indicators are sorted “globally”.

Next, we present an adaptive refinement algorithm, which uses the a posteriori error estimators developed in Section 3. The goal is the automatic adjustment of the temporal
and spatial discretizations in order to efficiently increase the accuracy. To obtain efficient
discretizations, it is essential to equilibrate the temporal and spatial discretization errors and
keep them balanced under further refinement. If the functional value \( J(u) \) is to be computed
to a given accuracy \( TOL \), this can be achieved by refining each discretization as long as the
the corresponding part of the error is larger than \( \frac{1}{2} TOL \). However, this might lead to an
inefficient algorithm, especially in the case when the temporal and spatial discretization error
are unbalanced in the beginning. Furthermore, the desired accuracy \( TOL \) might be too small
to be achieved with the given computational resources. In the following, we present an adaptive
algorithm, which balances the initial temporal and spatial discretization errors and keeps them
balanced during the further refinement process without having to prescribe a certain accuracy
\( TOL \). This leads us to an algorithm, which uses the given computational resources efficiently in
achieving the best accuracy. Therefore, the stopping criterion is based on reaching a prescribed
maximum number of degrees of freedom (determined by the given computer resources) rather
than on reaching the desired accuracy \( TOL \). Of course, these two different stopping criteria
can easily be exchanged by checking if

\[
|\eta| = |\eta_k + \eta_h| \leq TOL.
\]

As already mentioned, the goal of an efficient adaptive refinement algorithm has to be
the equilibrated reduction of the temporal and spatial discretization error. To this end, we
introduce an equilibration constant \( \kappa \geq 1 \) (usually \( \kappa \approx 3 \) in our numerical examples) and
proceed as described in Algorithm 1.

**Algorithm 1** Adaptive refinement algorithm

1: Choose an initial temporal and spatial discretization \( T_{k_0, h_0} \).
2: Set \( n = 0 \).
3: loop
4: Compute the primal and dual solution \( u_{k_n h_n} \) and \( z_{k_n h_n} \).
5: Evaluate the a posteriori error estimators \( \eta_{k_n} \) and \( \eta_{h_n} \).
6: if the maximum number of degrees of freedom is reached then
7: return
8: if \( |\eta_{k_n}| > \kappa |\eta_{h_n}| \) then
9: Adapt the temporal discretization.
10: else if \( |\eta_{h_n}| > \kappa |\eta_{k_n}| \) then
11: Adapt the spatial discretization.
12: else
13: Adapt the temporal and spatial discretization.
14: Increase \( n \).

**Remark 3.5.** The behavior of Algorithm 1 strongly depends on the choice of \( \kappa \). Choosing
\( \kappa \) too small results in a slower reduction of the overall discretization error because only the
temporal or spatial discretization is refined while the temporal and spatial discretization error
are actually of the same size. On the other hand, choosing \( \kappa \) too large makes the algorithm
inefficient because both discretizations are refined although the total discretization error is
dominated by only its temporal or spatial component. Numerical tests indicate that \( \kappa \approx 3 \) is a
good choice.
When refining a discretization the cells (or time intervals) to be refined are chosen using sets \( \Sigma_k \) or \( \Sigma_h \) of error indicators like the ones shown at the end of the previous section. Thus, we have to select subsets \( \Sigma_R \subseteq \Sigma_k \) or \( \Sigma_R \subseteq \Sigma_h \) indicating which cells (or time intervals) should be refined. As already noted in Remark 3.4, the selection of the spatial cells to be refined is done simultaneously on all meshes \( T^m \), \( m = 0, \ldots, M \). For the selection of the subsets \( \Sigma_R \) or \( \Sigma_R \), several standard approaches are available, e. g., “error balancing” or “fixed fraction strategies” (see Becker & Rannacher [25] or Bangerth & Rannacher [4]). However, for the computations presented in this paper a more sophisticated “look ahead” strategy has been used.

In the first step of this strategy, we compute a permutation \( (i_1, \ldots, i_N) \) of \( (1, \ldots, N) \) such that the corresponding local error indicators \( \eta_i \) on the current mesh are sorted in descending order according to their absolute value

\[
|\eta_{i_1}| \geq \cdots \geq |\eta_{i_N}|
\]

Then, the subset \( \Sigma_R = \{\eta_{i_1}, \ldots, \eta_{i_r}\} \subseteq \Sigma \) is chosen as coherent queue where the index \( r \) is given by

\[
r := \arg \min_{1 \leq s \leq N} \mathcal{E}(s) N(s)^{\beta}.
\]

Here, \( \mathcal{E}(s) \) is a prediction of the error for the refined discretization which is given by

\[
\mathcal{E}(s) := \sum_{i=1}^{N} |\eta_i| - \sum_{i=1}^{r} (1 - 2^{-\alpha})|\eta_i|.
\]

The parameter \( \alpha \) denotes the expected order of convergence, i.e., we assume that

\[
\sum_{K' \subset K} |\eta_{K'}| = 2^{-\alpha} |\eta_K|
\]

after refinement of a cell \( K \) into \( 2^d \) cells \( K' \). Further, \( N(s) \) is the number of degrees of freedom of the refined discretization and the parameter \( \beta \) is given as the quotient of the order of the finite element space and the dimension \( d \) of the discretized domain. The optimal value of \( r \) is determined by successively testing (20) for \( r = 1, \ldots, N \). More details on this approach can be found in Richter [34, 35].

4. NUMERICAL TESTS

In this section, we present some numerical results obtained by applying the proposed adaptive algorithm in combination with different temporal and spatial discretizations to the incompressible Navier-Stokes equations (1).

4.1. Example 1

As first example, we consider an artificial model problem on the two-dimensional unit square \( \Omega = (0,1)^2 \) and final time \( T = 1 \). We set \( \nu = 1 \) and choose the force \( f \) in such a way that
the exact solution \((v, p)^T\) is given by

\[
v(t, x) = \left( \sin(t) \sin^2(\pi x_1) \sin(\pi x_2) \cos(\pi x_2) \right),
\]

\[
p(t, x) = \sin(t) \sin(\pi x_1) \cos(\pi x_2) \cos(\pi x_2).
\]

The initial and boundary conditions are set to

\[
v = 0, \quad \text{in} \quad \{0\} \times \Omega,
\]

\[
v = 0, \quad \text{on} \quad (0, 1) \times \partial \Omega.
\]

We aim at computing the functional value

\[
J(u) = \frac{1}{2} \int_\Omega |v(1, x)|^2 \, dx
\]

at final time \(T = 1\). The exact value can be computed to be

\[
J(u) = \frac{3}{64} \sin^2(1) \approx 0.03319094148157365.
\]

At first, we present numerical evidence for the actual splitting of the total discretization error into a temporal and spatial component. Tables I and II show the independence of the temporal error estimator on the refinement of the spatial discretization and vice versa. This is an important feature in equilibrating both discretization errors during the adaptive algorithm presented in Section 3. Here and in the rest of this paper, \(N\) denotes the number of degrees of freedom of one spatial mesh while \(M\) denotes the number of time intervals. Further, we notice the very good agreement of the spatial error estimators between both temporal discretizations (columns three and four in Tables I and II) as well as the agreement of the temporal error estimators (columns five and six of these tables) when using either the cG(1) or the cG(2) discretization in space.

Table I. Independence of one part of the error estimator on the refinement of the other discretization: dG(0) discretization in time, cG(1) or cG(2) discretization in space

<table>
<thead>
<tr>
<th>(N)</th>
<th>(M)</th>
<th>(\eta_h)</th>
<th>(\eta_h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>243</td>
<td>40</td>
<td>cG(1)</td>
<td>cG(2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.8136 \cdot 10^{-04}</td>
<td>4.2374 \cdot 10^{-04}</td>
</tr>
<tr>
<td>867</td>
<td>40</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.1703 \cdot 10^{-04}</td>
<td>4.2887 \cdot 10^{-04}</td>
</tr>
<tr>
<td>3267</td>
<td>40</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.2620 \cdot 10^{-04}</td>
<td>4.2922 \cdot 10^{-04}</td>
</tr>
<tr>
<td>12675</td>
<td>40</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.2848 \cdot 10^{-04}</td>
<td>4.2924 \cdot 10^{-04}</td>
</tr>
<tr>
<td>49923</td>
<td>40</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.2905 \cdot 10^{-04}</td>
<td>4.2924 \cdot 10^{-04}</td>
</tr>
<tr>
<td>3267</td>
<td>10</td>
<td>1.9876 \cdot 10^{-04}</td>
<td>1.5554 \cdot 10^{-06}</td>
</tr>
<tr>
<td>3267</td>
<td>20</td>
<td>2.0636 \cdot 10^{-04}</td>
<td>1.6150 \cdot 10^{-06}</td>
</tr>
<tr>
<td>3267</td>
<td>40</td>
<td>2.1009 \cdot 10^{-04}</td>
<td>1.6442 \cdot 10^{-06}</td>
</tr>
<tr>
<td>3267</td>
<td>80</td>
<td>2.1194 \cdot 10^{-04}</td>
<td>1.6586 \cdot 10^{-06}</td>
</tr>
<tr>
<td>3267</td>
<td>160</td>
<td>2.1286 \cdot 10^{-04}</td>
<td>1.6656 \cdot 10^{-06}</td>
</tr>
</tbody>
</table>
Table II. Independence of one part of the error estimator on the refinement of the other discretization: dG(1) discretization in time, cG(1) or cG(2) discretization in space

<table>
<thead>
<tr>
<th>$N$</th>
<th>$M$</th>
<th>cG(1) $\eta_h$</th>
<th>cG(2) $\eta_h$</th>
<th>cG(1) $\eta_k$</th>
<th>cG(2) $\eta_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>243</td>
<td>40</td>
<td>$-5.4041 \cdot 10^{-07}$</td>
<td>$-5.7303 \cdot 10^{-07}$</td>
<td>$-5.6914 \cdot 10^{-07}$</td>
<td>$-5.7818 \cdot 10^{-07}$</td>
</tr>
<tr>
<td>867</td>
<td>40</td>
<td>$-5.7621 \cdot 10^{-07}$</td>
<td>$-5.7853 \cdot 10^{-07}$</td>
<td>$-5.7795 \cdot 10^{-07}$</td>
<td>$-5.7853 \cdot 10^{-07}$</td>
</tr>
<tr>
<td>3267</td>
<td>40</td>
<td>$-5.7839 \cdot 10^{-07}$</td>
<td>$-5.7853 \cdot 10^{-07}$</td>
<td>$-5.7839 \cdot 10^{-07}$</td>
<td>$-5.7853 \cdot 10^{-07}$</td>
</tr>
<tr>
<td>12675</td>
<td>40</td>
<td>$-5.7839 \cdot 10^{-07}$</td>
<td>$-5.7853 \cdot 10^{-07}$</td>
<td>$-5.7839 \cdot 10^{-07}$</td>
<td>$-5.7853 \cdot 10^{-07}$</td>
</tr>
<tr>
<td>49923</td>
<td>40</td>
<td>$-5.7839 \cdot 10^{-07}$</td>
<td>$-5.7853 \cdot 10^{-07}$</td>
<td>$-5.7839 \cdot 10^{-07}$</td>
<td>$-5.7853 \cdot 10^{-07}$</td>
</tr>
</tbody>
</table>

Table III and IV show the development of the total discretization error $J(u) - J(u_{kh})$ as well as the spatial and temporal error estimators $\eta_h$ and $\eta_k$, respectively, during one refinement cycle with local refinement of the temporal and spatial discretization using dynamic meshes for the cG(1)dG(0) and cG(2)dG(1) discretization. Here, $N_{\text{max}}$ denotes the number of degrees of freedom of the finest spatial mesh used whereas $N_{\text{tot}}$ denotes the total number of degrees of freedom of the space-time discretization. The last column shows the so-called “effectivity index” $I_{\text{eff}}$, which is given by

$$I_{\text{eff}} := \frac{J(u) - J(u_{kh})}{\eta_k + \eta_h}.$$ 

Looking at these tables, we observe $I_{\text{eff}} \approx 1$ for finer discretizations, which demonstrates the very good estimation of the discretization error. We also notice the equilibration of the temporal and spatial discretization error achieved during the refinement process.

Table III. Adaptive refinement on dynamic meshes with equilibration for the cG(1)dG(0) discretization

<table>
<thead>
<tr>
<th>$N_{\text{tot}}$</th>
<th>$N_{\text{max}}$</th>
<th>$M$</th>
<th>$\eta_h$</th>
<th>$\eta_k$</th>
<th>$J(u) - J(u_{kh})$</th>
<th>$I_{\text{eff}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2673</td>
<td>243</td>
<td>10</td>
<td>2.82 $\cdot 10^{-03}$</td>
<td>1.39 $\cdot 10^{-03}$</td>
<td>5.42 $\cdot 10^{-03}$</td>
<td>1.29</td>
</tr>
<tr>
<td>5655</td>
<td>867</td>
<td>12</td>
<td>8.26 $\cdot 10^{-04}$</td>
<td>8.33 $\cdot 10^{-04}$</td>
<td>1.96 $\cdot 10^{-04}$</td>
<td>1.18</td>
</tr>
<tr>
<td>18621</td>
<td>3267</td>
<td>14</td>
<td>2.25 $\cdot 10^{-04}$</td>
<td>5.04 $\cdot 10^{-04}$</td>
<td>8.05 $\cdot 10^{-04}$</td>
<td>1.10</td>
</tr>
<tr>
<td>91113</td>
<td>12435</td>
<td>18</td>
<td>6.32 $\cdot 10^{-05}$</td>
<td>2.71 $\cdot 10^{-04}$</td>
<td>3.57 $\cdot 10^{-04}$</td>
<td>1.07</td>
</tr>
<tr>
<td>162657</td>
<td>12435</td>
<td>26</td>
<td>6.07 $\cdot 10^{-05}$</td>
<td>1.41 $\cdot 10^{-04}$</td>
<td>2.08 $\cdot 10^{-04}$</td>
<td>1.03</td>
</tr>
<tr>
<td>767913</td>
<td>47859</td>
<td>34</td>
<td>1.94 $\cdot 10^{-05}$</td>
<td>8.51 $\cdot 10^{-05}$</td>
<td>1.03 $\cdot 10^{-04}$</td>
<td>0.98</td>
</tr>
<tr>
<td>1402389</td>
<td>47859</td>
<td>54</td>
<td>1.87 $\cdot 10^{-05}$</td>
<td>4.48 $\cdot 10^{-05}$</td>
<td>6.23 $\cdot 10^{-05}$</td>
<td>0.98</td>
</tr>
<tr>
<td>7419177</td>
<td>177627</td>
<td>82</td>
<td>6.36 $\cdot 10^{-06}$</td>
<td>2.63 $\cdot 10^{-05}$</td>
<td>3.18 $\cdot 10^{-05}$</td>
<td>0.97</td>
</tr>
</tbody>
</table>
Table IV. Adaptive refinement on dynamic meshes with equilibration for the cG(2)dG(1) discretization

<table>
<thead>
<tr>
<th>(N_{\text{tot}})</th>
<th>(N_{\text{max}})</th>
<th>(M)</th>
<th>(\eta_h)</th>
<th>(\eta_k)</th>
<th>(J(u) - J(u_{kh}))</th>
<th>(L_{\text{eff}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>5103</td>
<td>243</td>
<td>10</td>
<td>4.49 \times 10^{-04}</td>
<td>-1.71 \times 10^{-05}</td>
<td>3.84 \times 10^{-04}</td>
<td>0.89</td>
</tr>
<tr>
<td>6975</td>
<td>867</td>
<td>10</td>
<td>3.03 \times 10^{-05}</td>
<td>-2.89 \times 10^{-05}</td>
<td>4.12 \times 10^{-06}</td>
<td>2.81</td>
</tr>
<tr>
<td>27243</td>
<td>3267</td>
<td>12</td>
<td>1.68 \times 10^{-06}</td>
<td>-3.47 \times 10^{-06}</td>
<td>-2.43 \times 10^{-06}</td>
<td>1.35</td>
</tr>
<tr>
<td>106167</td>
<td>12675</td>
<td>14</td>
<td>1.12 \times 10^{-07}</td>
<td>-6.20 \times 10^{-07}</td>
<td>-5.12 \times 10^{-07}</td>
<td>1.01</td>
</tr>
<tr>
<td>233067</td>
<td>12675</td>
<td>20</td>
<td>1.21 \times 10^{-07}</td>
<td>-8.82 \times 10^{-08}</td>
<td>3.76 \times 10^{-08}</td>
<td>1.15</td>
</tr>
<tr>
<td>1028019</td>
<td>49923</td>
<td>24</td>
<td>1.01 \times 10^{-08}</td>
<td>-1.77 \times 10^{-08}</td>
<td>-8.78 \times 10^{-09}</td>
<td>1.15</td>
</tr>
<tr>
<td>7651347</td>
<td>197571</td>
<td>40</td>
<td>7.57 \times 10^{-10}</td>
<td>-2.56 \times 10^{-09}</td>
<td>-1.92 \times 10^{-09}</td>
<td>1.07</td>
</tr>
</tbody>
</table>

A comparison of different refinement strategies for the cG(1)dG(0) and cG(2)dG(1) discretization is presented in Figures 4 and 5, respectively. We use the following labeling:

- “uniform”: Uniform refinement is applied to the temporal and spatial discretization.
- “adaptive”: Adaptive refinement is applied to the temporal and spatial discretization together with the proposed equilibration strategy. The spatial mesh is fixed on the whole time interval.
- “dynamic”: Adaptive refinement is applied to the temporal and spatial discretization together with the proposed equilibration strategy. The spatial meshes may vary from time interval to time interval.

Even for this simple example with smooth solution, we achieve a reduction factor of about \(1/50 - 1/100\) in the degrees of freedom needed for reaching a certain accuracy when using adaptive refinement simultaneously in time and space compared to uniform refinement. In situations where the dynamics in space is stronger than here, one can achieve an even better reduction factor between the adaptive refinement on a fixed spatial mesh and dynamic meshes.

The size of the time steps obtained in the last iteration for the cG(1)dG(0) discretization is shown in Figure 6. The other discretization leads to similar adaptive refinement at the end of the time interval. This is not surprising since the functional \(J(\cdot)\) only acts at the final time \(T = 1\) and the solution is mainly driven by the force \(f\) and not by any intrinsic dynamics.

Finally, in Figure 7, we show a sequence of adaptively refined meshes obtained in the last iteration using dynamic meshes exemplarily for the cG(1)dG(0) discretization. We notice that the mesh is much more refined to the end of the time interval. Actually, the mesh is kept coarse and constant for \(t \in [0, 0.6]\).

4.2. Example 2

As a second example, we consider the benchmark configuration “Laminar Flow Around a Cylinder” (see Schäfer & Turek [36]). The configuration is shown in Figure 8. The aim of the simulation is the efficient computation of the mean drag coefficient.

The kinematic viscosity is set to \(\nu = 10^{-3}\) m² s⁻¹ while the density is given by \(\rho = 1\) kg m⁻³. As initial condition \(\mathbf{v}(0, \mathbf{x}) = 0\) m s⁻¹ is chosen. The inflow velocity at the left boundary is
prescribed as

\[ v_1(t, x) = \frac{6 \sin(\frac{\pi t}{8})}{(0.41 \text{ m})^2} x_2 (0.41 \text{ m} - x_2) \text{ m s}^{-1}, \quad v_2(t, x) = 0 \text{ m s}^{-1}. \]

At the outflow boundary on the right, we apply the artificial Neumann-type boundary condition (3) (see Heywood et al. [16]). On the upper and lower part of the boundary, we prescribe no-slip Dirichlet boundary conditions. The final time is set to \( T = 8 \text{ s} \). This setting leads to a Reynolds number \( \text{Re}(t) = \nu^{-1} \dot{U}(t) D \in [0, 100] \), for \( 0 \leq t \leq 8 \text{ s} \), based on the mean inflow velocity

\[ \dot{U}(t) = \frac{2}{3} v_1(t, 0 \text{ m}, 0.205 \text{ m}) = \sin(\frac{\pi t}{8 \text{ s}}) \text{ m s}^{-1}. \]
Figure 6. Adaptively determined time step size $k$ for the cG(1)dG(0) discretization

Figure 7. Spatial meshes at different time points obtained for the cG(1)dG(0) discretization

and the diameter of the obstacle $D = 0.1$ m.

**Remark 4.1.** For treating the nonhomogeneous Dirichlet boundary conditions the variational formulation has to be modified. Instead of seeking $u = (v, p)^T \in X$ satisfying

$$\int_I \left\{ (\partial_t v, \psi) + a(u)(\varphi) \right\} \, dt + (v(0) - v^0, \psi(0)) = 0 \quad \forall \varphi = (\psi, \chi)^T \in X,$$

we seek a solution $u = (v, p)^T \in (v^\Gamma, 0)^T + X$ satisfying (21) where $v^\Gamma$ is a divergence-free extension of the Dirichlet boundary data. This modification also influences the derivation of
the a posteriori error estimators because now the primal solution is no longer an admissible test function for the dual problem and hence in Lemma 5.1, below, we have

\[ L'(y_1)(e) \neq 0. \]

This leads to additional terms in the a posteriori error estimators, which are of the form

\[ \rho^*(u-z)(u-\tilde{u}_k), \quad \rho^*(u_k-z_k)(u_k-\tilde{u}_{kh}). \] (22)

These terms can be approximated using the same higher order reconstruction techniques as described in Section 3.2 to replace (22) by

\[ \rho^*(I_k u_{kh}, I_k z_{kh})(\Pi_k u_{kh}), \quad \rho^*(I_h u_{kh}, I_h z_{kh})(\Pi_h u_{kh}), \]

with some interpolation operators \( I_k \) and \( I_h \).

The mean drag coefficient is defined by

\[ J_d(u) := -\int_I \{(\partial_t v, \hat{\psi}_d) + a(u)(\hat{\phi}_d)\} \, dt, \] (23)

where \( \hat{\phi}_d = (\hat{s}_d, 0)^T \) fulfills

\[ \hat{\psi}_d|_S = (|I|^{-1} C, 0)^T, \quad \hat{\psi}_d|_{\partial \Omega \setminus S} = 0, \]

with the constant \( C = 2(\bar{\rho} \bar{U}^2 D)^{-1} \). Using the cG(2)dG(1) discretization with uniform refinement of the temporal and spatial discretizations, we obtain the values for the mean drag coefficient listed in Table V. Then, we employ the cG(1)dG(1) and cG(2)dG(1) discretization in combination with the adaptive Algorithm 1 to this problem. The results of these computations using adaptively refined spatial meshes which are kept constant over the whole time interval \( I = (0,8) \) are shown in Tables VI and VII. In these tables, we use the extrapolated value \( J_d(u) = 1.6072872 \) as reference solution. We observe how the spatial and the temporal discretization errors are equilibrated and kept balanced under further refinement. Further, we notice a quite good agreement of the estimated and the actual discretization error especially on finer discretizations, i.e., \( I_{eff} \approx 1 \).

Remark 4.2. The dG(0) discretization in time is not considered in this subsection due to its strong numerical dissipation, which makes the use of this discretization unfavorable for computing time periodic flows.
Remark 4.3. We do not use dynamic spatial meshes in this example for two reasons. On the one hand, the use of dynamic meshes leads to wrong approximations of the drag coefficient if no additional projection steps are applied each time the spatial mesh is changed, which would be rather costly (see Besier & Wollner [37] for a discussion of this problem). The other reason

Table V. Mean drag coefficient obtained by the cG(2)dG(1) discretization and uniform refinement

<table>
<thead>
<tr>
<th>$N$</th>
<th>$M$</th>
<th>$J_d(u_{hh})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2124</td>
<td>80</td>
<td>1.6178974</td>
</tr>
<tr>
<td>8088</td>
<td>160</td>
<td>1.5695421</td>
</tr>
<tr>
<td>31536</td>
<td>320</td>
<td>1.6048954</td>
</tr>
<tr>
<td>124512</td>
<td>640</td>
<td>1.6071242</td>
</tr>
<tr>
<td>494784</td>
<td>1280</td>
<td>1.6072465</td>
</tr>
<tr>
<td>extrapolated</td>
<td></td>
<td>1.6072872</td>
</tr>
</tbody>
</table>

Table VI. Mean drag coefficient: Adaptive refinement with equilibration for the cG(1)dG(1) discretization

<table>
<thead>
<tr>
<th>$N$</th>
<th>$M$</th>
<th>$\eta_h$</th>
<th>$\eta_k$</th>
<th>$J_d(u) - J_d(u_{hh})$</th>
<th>$I_{eff}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>582</td>
<td>80</td>
<td>$-2.99 \cdot 10^{-01}$</td>
<td>$2.16 \cdot 10^{-04}$</td>
<td>$1.35 \cdot 10^{-03}$</td>
<td>$-0.45$</td>
</tr>
<tr>
<td>1302</td>
<td>80</td>
<td>$-9.15 \cdot 10^{-02}$</td>
<td>$2.80 \cdot 10^{-04}$</td>
<td>$1.54 \cdot 10^{-03}$</td>
<td>$-1.69$</td>
</tr>
<tr>
<td>2280</td>
<td>80</td>
<td>$-3.99 \cdot 10^{-03}$</td>
<td>$3.26 \cdot 10^{-04}$</td>
<td>$7.03 \cdot 10^{-02}$</td>
<td>$-9.94$</td>
</tr>
<tr>
<td>5394</td>
<td>80</td>
<td>$3.29 \cdot 10^{-03}$</td>
<td>$3.96 \cdot 10^{-03}$</td>
<td>$3.64 \cdot 10^{-02}$</td>
<td>$4.41$</td>
</tr>
<tr>
<td>10998</td>
<td>120</td>
<td>$3.82 \cdot 10^{-02}$</td>
<td>$5.82 \cdot 10^{-03}$</td>
<td>$1.11 \cdot 10^{-02}$</td>
<td>$1.05$</td>
</tr>
<tr>
<td>25044</td>
<td>128</td>
<td>$3.46 \cdot 10^{-03}$</td>
<td>$7.86 \cdot 10^{-03}$</td>
<td>$1.69 \cdot 10^{-03}$</td>
<td>$1.08$</td>
</tr>
<tr>
<td>70146</td>
<td>256</td>
<td>$2.14 \cdot 10^{-04}$</td>
<td>$7.48 \cdot 10^{-04}$</td>
<td>$6.34 \cdot 10^{-04}$</td>
<td>$0.97$</td>
</tr>
<tr>
<td>70146</td>
<td>258</td>
<td>$1.14 \cdot 10^{-04}$</td>
<td>$7.32 \cdot 10^{-04}$</td>
<td>$8.15 \cdot 10^{-04}$</td>
<td>$0.96$</td>
</tr>
<tr>
<td>70146</td>
<td>516</td>
<td>$1.22 \cdot 10^{-04}$</td>
<td>$9.85 \cdot 10^{-05}$</td>
<td>$1.68 \cdot 10^{-04}$</td>
<td>$0.76$</td>
</tr>
<tr>
<td>235554</td>
<td>1032</td>
<td>$4.25 \cdot 10^{-05}$</td>
<td>$1.33 \cdot 10^{-05}$</td>
<td>$6.93 \cdot 10^{-05}$</td>
<td>$1.24$</td>
</tr>
</tbody>
</table>

Table VII. Mean drag coefficient: Adaptive refinement with equilibration for the cG(2)dG(1) discretization

<table>
<thead>
<tr>
<th>$N$</th>
<th>$M$</th>
<th>$\eta_h$</th>
<th>$\eta_k$</th>
<th>$J_d(u) - J_d(u_{hh})$</th>
<th>$I_{eff}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2124</td>
<td>80</td>
<td>$-6.19 \cdot 10^{-02}$</td>
<td>$9.37 \cdot 10^{-04}$</td>
<td>$-1.06 \cdot 10^{-02}$</td>
<td>$0.17$</td>
</tr>
<tr>
<td>5448</td>
<td>80</td>
<td>$2.40 \cdot 10^{-02}$</td>
<td>$8.20 \cdot 10^{-03}$</td>
<td>$5.50 \cdot 10^{-02}$</td>
<td>$1.71$</td>
</tr>
<tr>
<td>11148</td>
<td>160</td>
<td>$2.38 \cdot 10^{-03}$</td>
<td>$4.22 \cdot 10^{-03}$</td>
<td>$5.47 \cdot 10^{-03}$</td>
<td>$0.83$</td>
</tr>
<tr>
<td>27132</td>
<td>252</td>
<td>$-2.29 \cdot 10^{-04}$</td>
<td>$7.79 \cdot 10^{-04}$</td>
<td>$2.09 \cdot 10^{-04}$</td>
<td>$0.38$</td>
</tr>
<tr>
<td>27132</td>
<td>258</td>
<td>$-2.29 \cdot 10^{-04}$</td>
<td>$7.31 \cdot 10^{-04}$</td>
<td>$1.55 \cdot 10^{-04}$</td>
<td>$0.31$</td>
</tr>
<tr>
<td>27132</td>
<td>516</td>
<td>$-1.92 \cdot 10^{-04}$</td>
<td>$9.88 \cdot 10^{-05}$</td>
<td>$-4.06 \cdot 10^{-04}$</td>
<td>$4.33$</td>
</tr>
<tr>
<td>76656</td>
<td>1032</td>
<td>$-3.08 \cdot 10^{-05}$</td>
<td>$1.33 \cdot 10^{-05}$</td>
<td>$-2.52 \cdot 10^{-05}$</td>
<td>$1.44$</td>
</tr>
</tbody>
</table>
becomes clear if we have a look at Figure 9 where the adaptive spatial meshes corresponding to the last lines in Tables VI and VII are shown. We observe that in order to precisely determine the mean drag coefficient it is not necessary to resolve the whole von Kármán vortex street. Only a small recirculation zone behind the obstacle is strongly refined. Since the vortices in this region develop relatively early, we may conclude that allowing dynamic meshes would not provide a notable reduction in the degrees of freedom needed to reach the same accuracy as with adaptively refined but fixed spatial meshes. In virtue of the additional effort on dynamic meshes due to more frequent matrix reassembling and the additional projection steps, we reach the conclusion that the use of dynamic spatial meshes does not make sense in this particular flow example.

![Spatial meshes for the computation of the mean drag coefficient with different discretizations](image)

(a) $c^G(1)dG(1)$ discretization

(b) $c^G(2)dG(1)$ discretization

Figure 9. Spatial meshes for the computation of the mean drag coefficient with different discretizations

In Figure 10, we show the temporal evolution of the drag coefficient $c_d$ for both discretizations considered here. These values correspond to the finest adaptive discretization described in the last lines of Tables VI and VII with a relative error in the mean drag coefficient of less than $5 \cdot 10^{-5}$, which corresponds to an absolute error of less than $8 \cdot 10^{-5}$. We notice a perfect agreement of both curves. The corresponding adaptive spatial meshes are the ones already shown in Figure 9. Due to the higher order of the $c^G(2)$ discretization compared to the $c^G(1)$ discretization in space, the mesh in Figure 9(a) is stronger refined than the one in Figure 9(b).

Even though, in this example, we aim at efficiently computing the mean drag coefficient the maximum drag coefficient is also computed very accurately. A comparison of the results produced by the presented adaptive discretizations with the reference values of John [38] is given in Table VIII. Figures 11 and 12 show a comparison of different refinement strategies for the $c^G(1)dG(1)$ and $c^G(2)dG(1)$ discretizations, respectively. We use the same labeling as above. If we compare the number of degrees of freedom needed to reach a relative error of 1%, the required degrees of freedom can be reduced by a factor of $1/5 - 1/15$ using adaptive refinement with equilibration instead of uniform refinement, depending on the chosen discretization. This ratio increases when aiming at higher accuracies, e.g., to approximately $1/70$ for a relative error of 0.1% in the case of the $c^G(1)dG(1)$ discretization.
Figure 10. Temporal evolution of the drag coefficient \( c_d \) obtained by different discretizations with adaptive refinement.

Table VIII. Maximum drag coefficient: Comparison with reference values for different discretizations

<table>
<thead>
<tr>
<th>Discretization</th>
<th>( c_{d,\text{max}} )</th>
<th>( c_{d,\text{max}} - c_{d,\text{max}}^{(\text{ref})} )</th>
<th>( t(c_{d,\text{max}}) )</th>
<th>( t(c_{d,\text{max}}^{(\text{ref})}) - t(c_{d,\text{max}}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>cG(1)dG(1)</td>
<td>2.950921575</td>
<td>3.0 ( \cdot ) 10(^{-5} )</td>
<td>3.9375 s</td>
<td>3.2 ( \cdot ) 10(^{-4} )</td>
</tr>
<tr>
<td>cG(2)dG(1)</td>
<td>2.950914600</td>
<td>2.4 ( \cdot ) 10(^{-6} )</td>
<td>3.9375 s</td>
<td>3.2 ( \cdot ) 10(^{-4} )</td>
</tr>
</tbody>
</table>

5. APPENDIX

5.1. Abstract a posteriori error analysis

In the following, we provide the theoretical basis of the a posteriori error analysis described in Section 3. To this end, we state a slightly more general version of the abstract result of Becker & Rannacher [25], which underlies the Dual Weighted Residual (DWR) method for goal-oriented a posteriori error estimation and mesh adaptation.

**Lemma 5.1.** Let \( Y \) be a function space and \( L \) and \( \tilde{L} \) be three times Gâteaux differentiable functionals on \( Y \). We seek a stationary point \( y_1 \) of \( L \) on a subspace \( Y_1 \subseteq Y \): Find \( y_1 \in Y_1 \) such that

\[
L'(y_1)(\delta y_1) = 0 \quad \forall \delta y_1 \in Y_1.
\]  

(24)

This equation is approximated by a Galerkin method using the functional \( \tilde{L} \) on a subspace \( Y_2 \subseteq Y \). Hence, the discrete problem seeks \( y_2 \in Y_2 \) such that

\[
\tilde{L}'(y_2)(\delta y_2) = 0 \quad \forall \delta y_2 \in Y_2.
\]  

(25)
If the continuous solution \( y_1 \) additionally fulfills

\[
L'(y_1)(y_2) = 0,
\]

with the approximative solution \( y_2 \), we have the error representation

\[
L(y_1) - \tilde{L}(y_2) = \frac{1}{2} L'(y_2)(y_1 - \tilde{y}_2) + \frac{1}{2} (L - \tilde{L})'(y_2) (y_2 - y_2) + (L - \tilde{L})(y_2) + \mathcal{R},
\]

for arbitrary \( \tilde{y}_2 \in Y_2 \), where the remainder term \( \mathcal{R} \) is given in terms of \( e := y_1 - y_2 \) as

\[
\mathcal{R} = \frac{1}{2} \int_0^1 L'''(y_2 + se)(e,e,e) s(s - 1) \, ds.
\]
Proof. Using the main theorem of calculus, we have
\[ L(y_1) - \tilde{L}(y_2) = L(y_1) - L(y_2) + (L - \tilde{L})(y_2) = \int_0^1 L'(y_2 + se)(e) \, ds + (L - \tilde{L})(y_2). \]
Then, employing the trapezoidal rule
\[ \int_0^1 f(s) \, ds = \frac{1}{2} f(0) + \frac{1}{2} f(1) + \frac{1}{2} \int_0^1 f''(s) \cdot s \cdot (s - 1) \, ds \]
for approximating the integral, yields
\[ L(y_1) - \tilde{L}(y_2) = \frac{1}{2} L'(y_2)(e) + \frac{1}{2} L'(y_1)(e) + R + (L - \tilde{L})(y_2). \]
Because of (24) and (26), we have
\[ L'(y_1)(e) = 0. \]
Due to assertion (25), we may replace \( L'(y_2)(e) \) by
\[ L'(y_2)(y_1 - \tilde{y}_2) + L'(y_2)(\tilde{y}_2 - y_2) = L'(y_2)(y_1 - \tilde{y}_2) + (L - \tilde{L})(y_2)(\tilde{y}_2 - y_2) \]
for arbitrary \( \tilde{y}_2 \in Y_2 \). This completes the proof. \( \square \)

In order to apply the abstract error representation formula (27) to the present concrete situation, we recall from Section 3 the definitions of the Lagrangian functionals \( \mathcal{L} : X \times X \to \mathbb{R} \), \( \tilde{\mathcal{L}} : X_k^r \times X_k^r \to \mathbb{R} \), and \( \tilde{\mathcal{L}}_h : X_{kh}^{r,s} \times X_{kh}^{r,s} \to \mathbb{R} \):
\[
\mathcal{L}(u, z) := J(u) + \int_I \left\{ (f - \partial_t v, w) - a(u)(z) \right\} \, dt - (v(0) - v^0, w(0)),
\]
\[
\tilde{\mathcal{L}}(u_k, z_k) := J(u_k) + \sum_{m=1}^M \int_{I_m} \left\{ (f - \partial_t v_k, w_k) - a(u_k)(z_k) \right\} \, dt
\]
\[
\quad - \sum_{m=0}^{M-1} (|v_k|_{m, w_k^+} - (v_k^-_0 - v^0, w_k^-_0),
\]
\[
\tilde{\mathcal{L}}_h(u_{kh}, z_{kh}) := \tilde{\mathcal{L}}(u_{kh}, z_{kh}) - S_h(u_{kh}, z_{kh}),
\]
with
\[
S_h(u_{kh}, z_{kh}) := \sum_{m=1}^M \int_{I_m} s_h^m(u_{kh})(z_{kh}) \, dt,
\]
and the Lagrangian multipliers \( z = (w, q)^T, z_k = (w_k, q_k)^T, \) and \( z_{kh} = (w_{kh}, q_{kh})^T \). This leads us to the following result.

**Theorem 5.2.** Let \( (u, z)^T, (u_k, z_k)^T, \) and \( (u_{kh}, z_{kh})^T \) denote stationary points of \( \mathcal{L}, \tilde{\mathcal{L}}, \) and \( \tilde{\mathcal{L}}_h \) on different discretization levels, i. e.,
\[
L'(u, z)(\delta u, \delta z) = \mathcal{L}'(u, z)(\delta u, \delta z) = 0 \quad \forall (\delta u, \delta z)^T \in X \times X,
\]
\[
\tilde{\mathcal{L}}'(u_k, z_k)(\delta u_k, \delta z_k) = 0 \quad \forall (\delta u_k, \delta z_k)^T \in X_k^r \times X_k^r,
\]
\[
\tilde{\mathcal{L}}_h'(u_{kh}, z_{kh})(\delta u_{kh}, \delta z_{kh}) = 0 \quad \forall (\delta u_{kh}, \delta z_{kh})^T \in X_{kh}^{r,s} \times X_{kh}^{r,s}.
\]
Then, there hold the following error representation formulas for the discretization errors in time and space:

\[ J(u) - J(u_k) = \frac{1}{2} \tilde{L}(u_k, z_k)(u - u_k, z - z_k) + \mathcal{R}_k, \]

\[ J(u_k) - J(u_{kh}) = \frac{1}{2} \tilde{L}(u_{kh}, z_{kh})(u_k - u_{kh}, z_k - z_{kh}) + \frac{1}{2} S_h(u_{kh}, z_{kh}) + \mathcal{R}_h. \]

Here, \((\bar{u}_k, \bar{z}_k)^T \in X_k^r \times X_k^r\) and \((\bar{u}_{kh}, \bar{z}_{kh})^T \in X_{kh}^{r_s} \times X_{kh}^{r_s}\) can be chosen arbitrarily and the remainder terms \(\mathcal{R}_k\) and \(\mathcal{R}_h\) have the same structure as in Lemma 5.1.

**Proof.** Evaluating the Lagrangian functionals at the respective primal and dual solutions, there holds

\[ J(u) = \mathcal{L}(u, z) = \tilde{L}(u, z), \quad J(u_k) = \tilde{L}_h(u_k, z_k), \quad J(u_{kh}) = \tilde{L}_h(u_{kh}, z_{kh}). \tag{28} \]

Here, we have used that the first component \(v\) of \(u \in X\) is continuous and hence the additional terms in \(\tilde{L}\) vanish. Consequently, we can write

\[ J(u) - J(u_k) = \mathcal{L}(u, z) - \tilde{L}(u_k, z_k) = \tilde{L}(u, z) - \tilde{L}(u_k, z_k), \tag{29} \]

\[ J(u_k) - J(u_{kh}) = \tilde{L}(u_k, z_k) - \tilde{L}_h(u_{kh}, z_{kh}). \tag{30} \]

Next, we apply Lemma 5.1 with the settings

\[ L = \tilde{L}, \quad \tilde{L} = \tilde{L}, \quad Y_1 = X \times X, \quad Y_2 = X_k^r \times X_k^r \quad \text{for (29)}, \]

\[ L = \tilde{L}, \quad \tilde{L} = \tilde{L}_h, \quad Y_1 = X_k^r \times X_k^r, \quad Y_2 = X_{kh}^{r_s} \times X_{kh}^{r_s} \quad \text{for (30)}. \]

(i) In the second case, we have \(Y_2 \subseteq Y_1\) since \(X_{kh}^{r_s} \subseteq X_k^r\). Hence, we can take \(Y := Y_1\) and condition (26) is automatically fulfilled.

(ii) For the first case, we have to choose \(Y := Y_1 + Y_2\) since \(X_k^r \not\subseteq X\). Thus, we must check condition (26), which reads \(\tilde{L}'(u, z)(u_k, z_k) = 0\), or equivalently,

\[ \tilde{L}'_u(u, z)(u_k, z_k) = 0, \quad \tilde{L}'_z(u, z)(z_k) = 0. \tag{31} \]

We only give the proof of the second condition \(\tilde{L}'_z(u, z)(z_k) = 0\). The first one can be handled analogously. Due to the continuity of the first component of the continuous solution \(u\) with respect to time, the jump terms and the initial condition in \(\tilde{L}\) vanish for \(u \in X\). Hence, equation (31) may be rewritten in the form

\[ \sum_{m=1}^{M} \int_I \left\{ (f - \partial_t v, w_h) - a(u)(z_k) \right\} \, dt = 0. \]

By construction, the continuous solution \(u\) fulfills

\[ \int_I \{ (\partial_t v, \psi) + a(u)(\varphi) \} \, dt = \int_I (f, \psi) \, dt \quad \forall \varphi = (\psi, \chi)^T \in X. \tag{32} \]

Since \(X\) is dense in \(L^2(I, H_0^1(\Omega)^d \times L^2(\Omega)/\mathbb{R})\) with respect to the norm of \(L^2(I, H_0^1(\Omega)^d \times L^2(\Omega)/\mathbb{R})\) and since there are no time derivatives on \(\psi\) in (32), this equation also holds true for all \(\varphi \in L^2(I, H_0^1(\Omega)^d \times L^2(\Omega)/\mathbb{R})\). The inclusion \(z_k \in X_k^r \subseteq L^2(I, H_0^1(\Omega)^d \times L^2(\Omega)/\mathbb{R})\) then implies that condition (31) is fulfilled. Finally, the assertion of the theorem is a direct consequence of Lemma 5.1 applied to the separated errors (29) and (30). \(\square\)
5.2. Practical evaluation of error bounds

We exemplarily develop the concrete form of the approximate error estimators

$$\eta_k = \sum_{m=1}^{M} \eta_k^m,$$
$$\eta_h = \sum_{m=0}^{M} \eta_h^m = \sum_{m=0}^{M} \sum_{K \in T_h^m} \eta_h^m,$$

for the backward Euler variant of the cG(s)dG(0) scheme. To this end, temporal integrals involving \( u_{kh} \) and \( z_{kh} \) are approximated by the box rule whereas those involving \( I_k^{(1)} u_{kh} \) and \( J_k^{(1)} z_{kh} \) are evaluated using the trapezoidal rule. In the following, we use the notation \( f_m := f(t_m) \) and

$$U_m := (V_m, P_m)^T, \quad V_m := v_{kh,m}^-, \quad P_m := p_{kh,m}^-,$$
$$Z_m := (W_m, Q_m)^T, \quad W_m := w_{kh,m}^-,$$
$$d_k U_m := k_m^{-1}(U_m - U_{m-1}), \quad d_k V_m := k_m^{-1}(V_m - V_{m-1}),$$
$$d_k Z_m := k_m^{-1}(Z_m - Z_{m-1}), \quad d_k W_m := k_m^{-1}(W_m - W_{m-1}).$$

This results in the representation

$$\rho(u_{kh})(\Pi_k z_{kh}) = \sum_{m=1}^{M} \left\{ \frac{1}{2} k_m (f_{m-1}, (\Pi_k w_{kh})_{m-1}^+) + \frac{1}{2} k_m (f_m, (\Pi_k w_{kh})_{m}^-) \right\}$$
$$- \frac{1}{2} k_m a(u_{kh,m-1}^+, (\Pi_k z_{kh})_{m-1}^+) - \frac{1}{2} k_m a(u_{kh,m}^-, (\Pi_k z_{kh})_{m}^-)$$
$$- \left\{ [v_{kh}]_{m-1}, (\Pi_k w_{kh})_{m-1}^- \right\}$$
$$= \sum_{m=1}^{M} k_m^2 \left\{ \frac{1}{2} a(U_m)(d_k Z_m) + (d_k V_m, d_k W_m) - \frac{1}{2} (f_{m-1}, d_k W_m) \right\},$$

where we use that, by construction, \((\Pi_k w_{kh})_{m}^- = 0\), \(u_{kh,m-1}^+ = u_{kh,m}^-\), \((\Pi_k z_{kh})_{m-1}^+ = z_{kh,m-1}^- - z_{kh,m-1}^- = z_{kh,m-1}^- - z_{kh,m}^-\), and \([v_{kh}]_{m-1} = v_{kh,m-1}^+ - v_{kh,m-1}^- = v_{kh,m}^+ - v_{kh,m}^-\). Analogously, we obtain the representations

$$\rho^*(u_{kh}, z_{kh})(\Pi_k u_{kh}) = \sum_{m=1}^{M} \left\{ \frac{1}{2} a(U_m)(d_k U_m, Z_m) - \frac{1}{2} J_{m}^1(U_m)(d_k U_m) \right\},$$

and

$$\rho(u_{kh})(\Pi_h z_{kh}) = \sum_{m=1}^{M} k_m \left\{ (f_m, \Pi_h W_m) - a(U_m)(\Pi_h Z_m) \right\}$$
$$- (d_k V_m, \Pi_h W_m) \right\} - (V_0 - v_0^0, \Pi_h W_0),$$

$$\rho^*(u_{kh}, z_{kh})(\Pi_h u_{kh}) = \sum_{m=1}^{M} \left\{ J_{m}^1(U_m)(\Pi_h U_m) - a(U_m)(\Pi_h U_m, Z_m) \right\}$$
$$+ (\Pi_h V_m, d_k W_m) \right\} + J_{m}^2(U_M)(\Pi_h U_M) - (\Pi_h V_M, W_M).$$
Similar expressions are obtained for the cG(s)dG(1) discretization. Of course, quadrature rules of higher order have to be applied in this case in order to exactly evaluate the temporal integrals.

(i) The estimators $\eta^m_h$ for the temporal discretization error can directly be used for adapting the time step. They have the explicit form

$$
\eta^m_h = \frac{1}{2} k^2_m a(U_m) (d_k Z_m) + k^2_m (d_k V_m, d_k W_m) - \frac{1}{2} k^2_m (f_m-1, d_k W_m) \\
+ \frac{1}{2} k^2_m a(U_m) (d_k U_m, Z_m) - \frac{1}{2} k^2_m J'_1(U_m) (d_k U_m),
$$

for $m = 1, \ldots, M$.

(ii) For the spatial localization of the estimators $\eta^m_h$, there exist several strategies. The traditional one follows the standard procedure described, e.g., in Verfürth [1], Becker & Rannacher [32, 25], and Bangerth & Rannacher [4]. Starting from the representations

$$
\eta^0_h = -(V_0 - v^0, \Pi_h W_0),
$$

$$
\eta^m_h = \kappa_m \{ (f_m, \Pi_h W_m) - a(U_m)(\Pi_h Z_m) - (d_k V_m, \Pi_h W_m) + J'_1(U_m)(\Pi_h U_m) \\
- a(U_m)(\Pi_h U_m, Z_m) + (\Pi_h V_m, d_k W_m) \},
$$

$$
\eta^M_h = \kappa_m \{ (f_M, \Pi_h W_M) - a(U_M)(\Pi_h Z_M) - (d_k V_M, \Pi_h W_M) \\
+ J'_1(U_M)(\Pi_h U_M) - a(U_M)(\Pi_h U_M, Z_M) + (\Pi_h V_M, d_k W_M) \}
$$

we split all terms in $\eta^m_h$, $m = 0, \ldots, M$, into the contributions $\eta^m_{h,K}$ of the single mesh cells. Integration by parts in the second-order terms yields

$$
a(U_m)(\Pi_h Z_m) = \sum_{K \in T^m_h} \{ \nu(\nabla V_m, \nabla \Pi_h W_m)_K + (V_m \cdot \nabla V_m, \Pi_h W_m)_K \\
- (P_m, \nabla \cdot \Pi_h W_m)_K + (\nabla \cdot V_m, \Pi_h Q_m)_K \}
$$

$$
= \sum_{K \in T^m_h} \{ (-\nu\Delta V_m + V_m \cdot \nabla V_m + \nabla P_m, \Pi_h W_m)_K + (\nabla \cdot V_m, \Pi_h Q_m)_K \\
+ (\nu \partial_n V_m, \Pi_h W_m)_{\partial K} - (P_m, n \cdot \Pi_h W_m)_{\partial K} \},
$$

and rearranging terms,

$$
a(U_m)(\Pi_h Z_m) = \sum_{K \in T^m_h} \{ (-\nu\Delta V_m + V_m \cdot \nabla V_m + \nabla P_m, \Pi_h W_m)_K + (\nabla \cdot V_m, \Pi_h Q_m)_K \\
+ \frac{1}{2}[(\nu \partial_n V_m - P_m n), \Pi_h W_m]_{\partial K \setminus \partial \Omega} + (\nu \partial_n V_m - P_m n, \Pi_h W_m)_{\partial K \cap \partial \Omega} \},
$$

Analogously, we obtain

$$
a'(U_m)(\Pi_h U_m, Z_m) = \sum_{K \in T^m_h} \{ \nu(\nabla \Pi_h V_m, \nabla W_m)_K + (V_m \cdot \nabla \Pi_h V_m, W_m)_K \\
+ (\Pi_h V_m, \nabla V_m, W_m)_K + (\Pi_h P_m, \nabla \cdot W_m)_K - (\nabla \cdot \Pi_h V_m, Q_m)_K \}
$$

$$
= \sum_{K \in T^m_h} \{ ((\Pi_h V_m, -\nu\Delta W_m - V_m \cdot \nabla V_m + \nabla V_m^T W_m + \nabla Q_m)_K \\
+ (\Pi_h P_m, \nabla \cdot W_m)_K + (\Pi_h V_m, \nu \partial_n W_m)_{\partial K} - (\nabla \cdot V_m, \Pi_h V_m, W_m)_K \\
+ (n \cdot V_m, \Pi_h V_m \cdot W_m)_{\partial K} - (n \cdot \Pi_h V_m, Q_m)_{\partial K} \},
$$
and again rearranging terms,

\[
a'(U_m)(\Pi_h U_m, \mathbf{Z}_m) = \sum_{K \in \mathcal{T}_h} \left\{ (\Pi_h V_m, -\nu \Delta W_m - V_m \cdot \nabla W_m + \nabla V_m^T W_m + \nabla Q_m)_{\Omega} \right. \\
+ (\Pi_h P_m, \nabla \cdot W_m)_{\Omega} - (\nabla \cdot V_m W_m, \Pi_h V_m)_{\Omega} \\
+ \frac{1}{2} (\Pi_h V_m, [\nu \partial_n W_m - Q_m n + n \cdot V_m W_m])_{\partial K \setminus \partial \Omega} \\
\left. + (\Pi_h V_m, \nu \partial_n W_m - Q_m n + n \cdot V_m W_m)_{\partial K \cap \partial \Omega} \right\}.
\]

Here \((\cdot, \cdot)_{\Omega}\) and \((\cdot, \cdot)_{\partial K}\) denote the $L^2$ scalar product over a mesh cell \(K\) and its boundary, respectively, and \([\cdot]\) is the jump of the quantity enclosed by the brackets across the interelement boundary. Since here we use continuous pressure approximations the pressure jump terms vanish, \([P_m n] \equiv [Q_m n] \equiv 0\). The same holds for the normal flux jump terms, \([n \cdot V_m] \equiv 0\).

From this, we can extract the cellwise spatial error indicators. To simplify notation, we assume from now on that \(J_2(\cdot) = 0\) and that \(J_1(\cdot)\) is given in the form \(J_1(u(t)) := (j(t), v(t))\) with some density function \(j \in L^2(I, L^2(\Omega)^d)\). In this case, we have

\[
J'_1(U_m)(\Pi_h U_m) = (j_m, \Pi_h V_m).
\]

Then, collecting all the terms derived above yields the following explicit representation of the cellwise spatial error indicators:

\[
\eta_{h,K}^m = k_m \{(f_m + \nu \Delta V_m - V_m \cdot \nabla V_m - \nabla P_m - d_k V_m, \Pi_h W_m)_{\Omega} - (\nabla \cdot V_m, \Pi_h Q_m)_{\Omega} \\
- \frac{1}{2} ([\nu \partial_n V_m - P_m n], \Pi_h W_m)_{\partial K \setminus \partial \Omega} - ([\nu \partial_n V_m - P_m n, \Pi_h W_m]_{\partial K \cap \partial \Omega} \\
+ (\Pi_h V_m, j_m + \nu \Delta W_m + V_m \cdot \nabla W_m - \nabla V_m^T W_m - \nabla Q_m + d_k W_m)_{K} \\
- (\Pi_h P_m, \nabla \cdot W_m)_{\Omega} + (\nabla \cdot V_m W_m, \Pi_h V_m)_{\Omega} \\
- \frac{1}{2} (\Pi_h V_m, [\nu \partial_n W_m - Q_m n + n \cdot V_m W_m])_{\partial K \setminus \partial \Omega} \\
- (\Pi_h V_m, \nu \partial_n W_m - Q_m n + n \cdot V_m W_m)_{\partial K \cap \partial \Omega} \},
\]

for \(m = 1, \ldots, M-1\). Similar representations are obtained for \(\eta_{h,K}^0\) and \(\eta_{h,K}^M\).

The practical evaluation of these local error indicators requires the computation of equation residuals of the discrete solution within cells and certain jumps across cell interfaces. This may be costly depending on the particular data structure employed in the software realization of the adaptive discretization. In an alternative approach introduced in Braack & Ern [33] the evaluation of equation residuals and interface jumps is avoided by replacing those by nodal point-oriented terms. The numerical results presented in this paper have been obtained using this simpler form of the error indicators, which, under certain regularity conditions, is asymptotically equivalent to the original one. For the rather technical details of the derivation of these nodal point-oriented error indicators, we refer to Schmich & Vexler [9] and especially to Schmich [8], for the case of the Navier-Stokes equations.

REFERENCES


30. Meidner D. Adaptive space-time finite element methods for optimization problems governed by nonlinear...
31. Carstensen C, Verfürth R. Edge residuals dominate a posteriori error estimates for low order finite element
32. Becker R, Rannacher R. A feedback approach to error control in finite element methods: Basic analysis
34. Richter T. Funktionalorientierte Gitteroptimierung bei der Finite-Elemente-Approximation elliptischer
35. Richter T. Parallel multigrid method for adaptive finite elements with application to 3d flow problems.
36. Schäfer M, Turek S. Benchmark computations of laminar flow around a cylinder. *Flow Simulation with
High-Performance Computer II, Notes on Numerical Fluid Mechanics*, vol. 52, Hirschel EH (ed.), Vieweg:
Braunschweig Wiesbaden, 1996; 547–566.
37. Besier M, Wollner W. On the pressure approximation in nonstationary incompressible flow simulations on
38. John V. Reference values for drag and lift of a two-dimensional time-dependent flow around a cylinder.