Goal-Oriented Adaptivity in Space-Time Finite Element Simulations of Nonstationary Incompressible Flows

Michael Besier

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

Abstract

Subject of this paper is the development of an a posteriori error estimator for nonstationary incompressible flow problems. The error estimator is computable and able to assess the temporal and spatial discretization errors separately. Thereby, the error is measured in an arbitrary quantity of interest because measuring errors in global norms is often of minor importance in practical applications. The basis for this is a finite element discretization in time and space. The techniques presented here also provide local error indicators which are used to adaptively refine the temporal and spatial discretization. A key ingredient in setting up an efficient discretization method is balancing the error contributions due to temporal and spatial discretization. To this end, a quantitative assessment of the individual discretization errors is required. The described method is validated by an established Navier-Stokes benchmark.

Keywords: nonstationary incompressible flows, space-time finite element methods, goal-oriented a posteriori error estimation, adaptivity
2010 MSC: 65M50, 65M60, 76D05, 76M10

1. Introduction

This work is devoted to the development of efficient discretization techniques for numerically solving nonstationary incompressible flow problems.

Email address: michael.besier@iwr.uni-heidelberg.de (Michael Besier)

Since in contrast to stationary problems we have to deal with the discretization in time as well as in space, one of the main topics in setting up such an efficient algorithm is to obtain quantitative information about the temporal and spatial discretization error. This is a key ingredient because within an efficient algorithm one has to decide which discretization has to be refined to reduce the discretization error in the most efficient way.

Adaptive methods are widely used in the context of finite element discretizations of partial differential equations, see, for example, [1] or [2] for an overview. In [3] and [4], adaptive time-stepping methods for the incompressible Navier-Stokes equations are discussed. However, a uniform spatial discretization is applied. On the other hand, in [5], a spatially adaptive (and uniform in 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* sometimes is not very efficient since in flow problems one is often only 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 problem is the efficient computation of this single number. This quantity might, for instance, be the mean drag- or lift-coefficient of an obstacle which is surrounded by the fluid. In this case, the efficiency of an algorithm for numerically computing this quantity has to be measured by means 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 author, see [6]. They are an extension of the methodology developed in [7] to nonstationary flow problems allowing for the simultaneous adaptation of the temporal and spatial discretization. We will derive a posteriori error estimators which quantitatively assess the discretization error measured in the quantity of interest and separate the influence of the temporal and the spatial discretization. This separation will allow us to set up an efficient algorithm for the adaptive refinement of the temporal and the spatial discretization.

The key to rigorous a posteriori error estimation is a coupled variational formulation of the underlying equations. It allows to apply Galerkin finite element methods not only for the discretization in space, but also for the discretization in time. The use of space-time finite element discretizations enables the application of residual based a posteriori error estimation. Space-

time Galerkin methods have already been applied successfully to the simulation of incompressible flows, see, for example, [8], [9], [10], or [11] as well as [12] (referred to as *General Galerkin G2*). While the first references do not consider adaptivity, in [12] also an adaptive algorithm for nonstationary flow problems based on a posteriori error estimation is developed. However, the author does not separate the temporal and spatial discretization error. Instead, the temporal refinement is linked to the spatial refinement.

In this paper, we consider nonstationary incompressible flows described by the incompressible Navier-Stokes equations, given in the dimensionless form as

$$\begin{aligned} \partial_t \boldsymbol{v} - \nu \Delta \boldsymbol{v} + (\boldsymbol{v} \cdot \nabla)\boldsymbol{v} + \nabla p &= \boldsymbol{f} && \text{in } I \times \Omega, \\ \nabla \cdot \boldsymbol{v} &= 0 && \text{in } I \times \Omega, \\ \boldsymbol{v}(0) &= \boldsymbol{v^0} && \text{in } \Omega \end{aligned} \tag{1}$$

with a time interval $I = (0, T)$, computational domain $\Omega \subseteq \mathbb{R}^d$, $d \in \{ 2, 3 \}$, kinematic viscosity ν, volume forces $\boldsymbol{f}$, and initial values $\boldsymbol{v^0}$. These equations have to be supplemented by appropriate boundary conditions. For sake of simplicity, we will assume no-slip Dirichlet boundary conditions.

The corresponding variational formulation reads as follows: For $\boldsymbol{f} \in L^2(I, H^{-1}(\Omega)^d)$ and $\boldsymbol{v^0} \in L^2(\Omega)^d$ find $\boldsymbol{u} \coloneqq (\boldsymbol{v}, p)^T \in X$ such that

$$\begin{aligned} \int_I \left\{ (\partial_t \boldsymbol{v}, \boldsymbol{\psi}) + a(\boldsymbol{u})(\boldsymbol{\varphi}) \right\} \mathrm{d}t + (\boldsymbol{v}(0) - \boldsymbol{v^0}, \boldsymbol{\psi}(0)) \\ = \int_I (\boldsymbol{f}, \boldsymbol{\psi}) \, \mathrm{d}t \quad \forall \boldsymbol{\varphi} \coloneqq (\boldsymbol{\psi}, \chi)^T \in X \end{aligned} \tag{2}$$

with the semi-linear form

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

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

$$\begin{aligned} X \coloneqq \Big\{ \boldsymbol{u} = (\boldsymbol{v}, p)^T \;\Big|\; & \boldsymbol{v} \in L^2(I, H_0^1(\Omega)^d), \; \partial_t \boldsymbol{v} \in L^2(I, H^{-1}(\Omega)^d), \\ & p \in L^2(I, L^2(\Omega)/\mathbb{R}) \Big\}. \end{aligned}$$

For questions on existence and uniqueness of solutions, we refer to [13].

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, there will be some part Γ_{out} of the boundary representing an outlet. Then, we apply natural boundary conditions on Γ_{out}:

$$\nu \partial_{\boldsymbol{n}} \boldsymbol{v} - p\boldsymbol{n} = \boldsymbol{0}.$$

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

$$\boldsymbol{v} \in \left\{ \boldsymbol{v} \in H^1(\Omega)^d \;\middle|\; \boldsymbol{v}\big|_{\partial\Omega\setminus\Gamma_{\text{out}}} = \boldsymbol{0} \right\}, \quad p \in L^2(\Omega).$$

For more information on this free outflow boundary condition as well as results concerning existence and uniqueness of solutions, we refer to [14].

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

$$J(\boldsymbol{u}) = \int_0^T J_1(\boldsymbol{u}(t))\ \mathrm{d}t + J_2(\boldsymbol{u}(T)),$$

where of course J_1 or J_2 may be zero. This choice covers the following two typical situations: The quantity of interest is a mean value of a given functional (J_1), or one is interested in a terminal value $J_2(\boldsymbol{u}(T))$.

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

$$J(\boldsymbol{u}) - J(\boldsymbol{u_{kh}}) \approx \eta_k + \eta_h,$$

where η_k describes the error due to the discretization in time and η_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 (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. The error estimates assess separately the error due to the discretization in time and in space and are obtained by using the solution of a (linear) dual equation. In Section 4, we describe the numerical realization of the derived error estimates and an adaptive algorithm for successive improvement of the accuracy. In Section 5, numerical results are presented, illustrating the behavior of the method.

2. Discretization

In this section, we describe the discretization of the weak formulation of the incompressible Navier-Stokes equations (2). The discretization in space as well as in time will be done by means of Galerkin finite element methods.

In the following subsection, we present the semi-discretization in time by *discontinuous Galerkin* (dG) methods. Subsection 2.2 then deals with the discretization in space of the arising semi-discrete problems. This is done by continuous Galerkin finite element methods. For technical reasons, we use piecewise polynomial functions of the same degree for the velocity and the pressure component. Hence, the Babuška-Brezzi stability condition is not fulfilled. Therefore, we have to apply stabilization techniques. This is done by the so-called *local projection stabilization* (LPS) and is described in Subsection 2.3 in more detail.

2.1. Discretization in time

To introduce the semi-discretizations in time, we partition the time interval $\bar{I} = [0, T]$ into

$$\bar{I} = \{\, 0 \,\} \cup I_1 \cup \cdots \cup I_m \cup \cdots \cup I_M$$

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 setting $k\big|_{I_m} := k_m$ for $m = 1, \ldots, M$.

Using the subintervals I_m, let us define the following semi-discrete spaces X_k^r for $r \in \mathbb{N}_0$:

$$\begin{aligned} X_k^r := \Big\{\, \boldsymbol{u_k} = (\boldsymbol{v_k}, p_k)^T \;\Big|\; & \boldsymbol{v_k}(0) \in L^2(\Omega)^d,\ \boldsymbol{v_k}\big|_{I_m} \in \mathcal{P}_r(I_m, H_0^1(\Omega)^d), \\ & p_k\big|_{I_m} \in \mathcal{P}_r(I_m, L^2(\Omega)/\mathbb{R}),\ m = 1, \ldots, M \,\Big\} \\ \subseteq\ & L^2(I, H_0^1(\Omega)^d \times L^2(\Omega)/\mathbb{R}), \end{aligned}$$

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

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

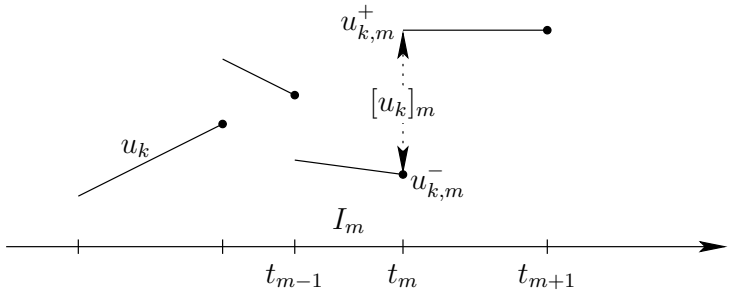


Figure 1: Notation of discontinuous functions u_k in the case $r = 1$

Thus, $u^+_{k,m}$ is the limit "from above" while $u^-_{k,m}$ denotes the limit "from below". $[u_k]_m$ then is the "jump" of $u_k(t)$ at $t = t_m$, see Figure 1.

The dG(r) semi-discretization of the incompressible Navier-Stokes equations (2) reads: Find $\boldsymbol{u_k} = (\boldsymbol{v_k}, p_k)^T \in X_k^r$ such that

$$\sum_{m=1}^{M} \int_{I_m} \left\{ (\partial_t \boldsymbol{v_k}, \boldsymbol{\psi}) + a(\boldsymbol{u_k})(\boldsymbol{\varphi}) \right\} \, \mathrm{d}t + \sum_{m=0}^{M-1} ([\boldsymbol{v_k}]_m, \boldsymbol{\psi}_m^+) + (\boldsymbol{v}_{\boldsymbol{k},0}^-, \boldsymbol{\psi}_0^-)$$
$$= \int_I (\boldsymbol{f}, \boldsymbol{\psi}) \, \mathrm{d}t + (\boldsymbol{v^0}, \boldsymbol{\psi}_0^-) \quad \forall \boldsymbol{\varphi} = (\boldsymbol{\psi}, \chi)^T \in X_k^r. \quad (3)$$

Remark 2.1. Due to the discontinuity of the test functions, the dG(r) discretizations decouple into time stepping schemes. 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 some implicit Runge-Kutta method.

2.2. Discretization in space

In this subsection, 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., [15]. A mesh consists of quadrilateral or hexahedral cells K which form a non-overlapping cover of the computational domain $\Omega \subseteq \mathbb{R}^d$. The corresponding mesh is denoted by $\mathcal{T}_h = \{ K \}$, where the discretization parameter h is defined as a cellwise constant function by setting $h\big|_K = h_K$ with the diameter h_K of the cell K.

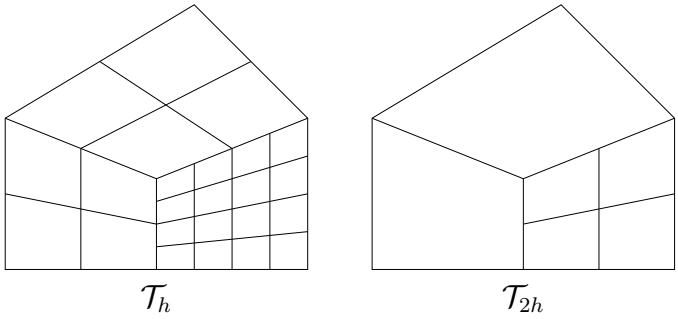


Figure 2: Two-dimensional mesh $\mathcal{T}_h$ (with hanging nodes) organized in a patch-wise manner with corresponding coarser mesh $\mathcal{T}_{2h}$

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 corresponding to these irregular nodes and the value of a finite element function is determined by pointwise interpolation, see [16] for more details.

On the mesh $\mathcal{T}_h$, we construct a conforming finite element space $V_h^s \subset H^1(\Omega)$ in a standard way:

$$V_h^s \coloneqq \left\{ v \in C(\overline{\Omega}) \;\middle|\; v\big|_K \in \mathcal{Q}_s(K) \text{ for } K \in \mathcal{T}_h \right\} \subseteq H^1(\Omega).$$

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

$$\hat{\mathcal{Q}}_s(\hat{K}) = \operatorname{span}\left\{ \prod_{j=1}^{d} x_j^{\alpha_j} \;\middle|\; \alpha_j \in \{\, 0, \ldots, s \,\} \right\}.$$

In addition, we will require that the mesh is organized in a patch-wise manner. That is, $\mathcal{T}_h$ is obtained by uniform refinement of a coarser mesh $\mathcal{T}_{2h}$, such that we can always combine four ($d = 2$) or eight ($d = 3$) adjacent cells of $\mathcal{T}_h$ to obtain one cell of $\mathcal{T}_{2h}$. 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 time steps k_m are kept constant in space. To

this end, we associate with each time point t_m a mesh $\mathcal{T}_h^m$ and corresponding (spatial) finite element spaces $V_h^{s_{\boldsymbol{v}},m}$ and $V_h^{s_p,m}$. We then define the following space-time finite element space:

$$\begin{aligned} X_{kh}^{r,s} \coloneqq \Big\{ \boldsymbol{u_{kh}} = (\boldsymbol{v_{kh}}, p_{kh})^T \;\Big|\; \boldsymbol{v_{kh}}(0) \in (H_h^0)^d,\ \boldsymbol{v_{kh}}\big|_{I_m} \in \mathcal{P}_r(I_m, (H_h^m)^d), \\ p_{kh}\big|_{I_m} \in \mathcal{P}_r(I_m, L_h^m),\ m = 1, \dots, M \Big\} \\ \subseteq L^2(I, H_0^1(\Omega)^d \times L^2(\Omega)/\mathbb{R}), \end{aligned}$$

where

$$H_h^m \coloneqq V_h^{s_{\boldsymbol{v}},m} \cap H_0^1(\Omega) \quad \text{and} \quad L_h^m \coloneqq V_h^{s_p,m} \cap L^2(\Omega)/\mathbb{R}.$$

Because of the conformity of H_h^m and L_h^m, we have $X_{kh}^{r,s} \subseteq X_k^r$.

Then, the cG(s)dG(r) formulation of problem (2) reads: Find $\boldsymbol{u_{kh}} = (\boldsymbol{v_{kh}}, p_{kh})^T \in X_{kh}^{r,s}$ such that

$$\begin{aligned} \sum_{m=1}^{M} \int_{I_m} \left\{ (\partial_t \boldsymbol{v_{kh}}, \boldsymbol{\psi}) + a(\boldsymbol{u_{kh}})(\boldsymbol{\varphi}) \right\} \mathrm{d}t + \sum_{m=0}^{M-1} ([\boldsymbol{v_{kh}}]_m, \boldsymbol{\psi}_m^+) + (\boldsymbol{v}_{\boldsymbol{kh},0}^-, \boldsymbol{\psi}_0^-) \\ = \int_I (\boldsymbol{f}, \boldsymbol{\psi})\ \mathrm{d}t + (\boldsymbol{v^0}, \boldsymbol{\psi}_0^-) \quad \forall \boldsymbol{\varphi} = (\boldsymbol{\psi}, \chi)^T \in X_{kh}^{r,s}. \quad (4) \end{aligned}$$

Remark 2.3. The notation cG(s)dG(r), representing a space-time finite element discretization with continuous piecewise polynomials of degree s in space and discontinuous piecewise polynomials of degree r in time, is taken from [17].

2.3. Stabilization

The fully discrete formulation 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., $s_{\boldsymbol{v}} = s_p = s$, which are favorable from the implementational point of view, do not fulfill this condition. Therefore we either have to use mixed finite element methods like the Taylor-Hood element (see [18]) or add stabilization terms. For implementational reasons, we apply equal-order trial spaces and apply the local projection stabilization, see, e. g., [19, 20].

To give a precise definition of the modified fully discrete formulations, we introduce a spatial interpolation operator $I_h : V_h^{s,m} \to \widetilde{V}_h^{s,m}$ into a subspace

$\widetilde{V}_h^{s,m} \subseteq V_h^{s,m}$ which is given as

$$\widetilde{V}_h^{s,m} \coloneqq \begin{cases} V_{2h}^{1,m} & \text{for } s = 1, \\ V_h^{1,m} & \text{for } s = 2. \end{cases}$$

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 $\pi : V_h^{s,m} \to V_h^{s,m}$ by

$$\pi \coloneqq \mathrm{id} - I_h.$$

The filtering operator $\boldsymbol{\pi} : (V_h^{s,m})^d \to (V_h^{s,m})^d$ is defined analogously componentwise. Let us further extend these operators in time by defining point-wise

$$(\pi p_{kh})(t) \coloneqq \pi p_{kh}(t) \quad \text{and} \quad (\boldsymbol{\pi v_{kh}})(t) \coloneqq \boldsymbol{\pi} \boldsymbol{v_{kh}}(t).$$

This allows us to state the following modified fully discrete formulation of problem (2): Find $\boldsymbol{u_{kh}} = (\boldsymbol{v_{kh}}, p_{kh})^T \in X_{kh}^{r,s}$ such that

$$\begin{aligned} \sum_{m=1}^{M} \int_{I_m} \left\{ (\partial_t \boldsymbol{v_{kh}}, \boldsymbol{\psi}) + a(\boldsymbol{u_{kh}})(\boldsymbol{\varphi}) + s_h^m(\boldsymbol{u_{kh}})(\boldsymbol{\varphi}) \right\} \mathrm{d}t + \sum_{m=0}^{M-1} ([\boldsymbol{v_{kh}}]_m, \boldsymbol{\psi}_m^+) \\ + (\boldsymbol{v}_{\boldsymbol{kh},0}^-, \boldsymbol{\psi}_0^-) = \int_I (\boldsymbol{f}, \boldsymbol{\psi}) \, \mathrm{d}t + (\boldsymbol{v^0}, \boldsymbol{\psi}_0^-) \quad \forall \boldsymbol{\varphi} = (\boldsymbol{\psi}, \chi)^T \in X_{kh}^{r,s}. \end{aligned} \tag{5}$$

Here, the additional terms are given by

$$s_h^m(\boldsymbol{u})(\boldsymbol{\varphi}) \coloneqq \sum_{K \in \mathcal{T}_h^m} \left\{ (\nabla \pi p, \alpha_{K,m} \nabla \pi \chi)_K + ((\boldsymbol{v} \cdot \nabla) \boldsymbol{\pi v}, \delta_{K,m} (\boldsymbol{v} \cdot \nabla) \boldsymbol{\pi \psi})_K \right\}$$

with $\boldsymbol{u} = (\boldsymbol{v}, p)^T$ and $\boldsymbol{\varphi} = (\boldsymbol{\psi}, \chi)^T$. The cell-wise stabilization parameters $\alpha_{K,m}$ and $\delta_{K,m}$ are given as

$$\alpha_{K,m} = \alpha_0 \frac{h_K^2}{6\nu + h_K \left\| \boldsymbol{v_{kh}} \right\|_K} \quad \text{and} \quad \delta_{K,m} = \delta_0 \frac{h_K^2}{6\nu + h_K \left\| \boldsymbol{v_{kh}} \right\|_K + \frac{h_K}{k_m}}$$

with some constants α_0 and δ_0. For details on the choice of these parameters, we refer, for instance, to [21] or [22]. In our computations, we chose $\alpha_0 = \delta_0 = 0.3$.

3. A posteriori error estimation

This section is dedicated to the development of an a posteriori error estimator which measures the discretization error in a functional J. The error estimator developed here is an extension of concepts already published in [7]. 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 during which the temporal and spatial discretization errors are balanced and simultaneously decreased.

To this end, we state a slightly more general version of the abstract result of [23]:

Lemma 3.1. *Let Y be a function space and L and $\widetilde{L}$ 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*

$$L'(y_1)(\delta y_1) = 0 \quad \forall \delta y_1 \in Y_1. \tag{6}$$

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

$$\widetilde{L}'(y_2)(\delta y_2) = 0 \quad \forall \delta y_2 \in Y_2. \tag{7}$$

If the continuous solution y_1 additionally fulfills

$$L'(y_1)(y_2) = 0 \tag{8}$$

with the approximative solution y_2, we have for arbitrary $\tilde{y}_2 \in Y_2$ the error representation

$$L(y_1) - \widetilde{L}(y_2) = \frac{1}{2}L'(y_2)(y_1 - \tilde{y}_2) + \frac{1}{2}(L - \widetilde{L})'(y_2)(\tilde{y}_2 - y_2) + (L - \widetilde{L})(y_2) + \mathcal{R}, \tag{9}$$

where the remainder term $\mathcal{R}$ is given by means of $e := y_1 - y_2$ as

$$\mathcal{R} = \frac{1}{2}\int_0^1 L'''(y_2 + se)(e,e,e) \cdot s \cdot (s-1)\, \mathrm{d}s.$$

Proof. We write by the main theorem of calculus

$$\begin{aligned} L(y_1) - \widetilde{L}(y_2) &= L(y_1) - L(y_2) + (L - \widetilde{L})(y_2) \\ &= \int_0^1 L'(y_2 + se)(e)\ \mathrm{d}s + (L - \widetilde{L})(y_2). \end{aligned}$$

Using the trapezoidal rule

$$\int_0^1 f(s)\ \mathrm{d}s = \frac{1}{2}f(0) + \frac{1}{2}f(1) + \frac{1}{2}\int_0^1 f''(s) \cdot s \cdot (s-1)\ \mathrm{d}s$$

for approximating the integral, supplies

$$L(y_1) - \widetilde{L}(y_2) = \frac{1}{2}L'(y_2)(e) + \frac{1}{2}L'(y_1)(e) + \mathcal{R} + (L - \widetilde{L})(y_2).$$

Because of (6) and (8), we have

$$L'(y_1)(e) = 0.$$

Due to assertion (7), we may replace $L'(y_2)(e)$ by

$$L'(y_2)(y_1 - \tilde{y}_2) + L'(y_2)(\tilde{y}_2 - y_2) = L'(y_2)(y_1 - \tilde{y}_2) + (L - \widetilde{L})'(y_2)(\tilde{y}_2 - y_2)$$

for arbitrary $\tilde{y}_2 \in Y_2$. This completes the proof. □

As mentioned in Section 1, we assume the functional J to be given in the form

$$J(\boldsymbol{u}) = \int_0^T J_1(\boldsymbol{u}(t))\ \mathrm{d}t + J_2(\boldsymbol{u}(T))$$

where of course J_1 or J_2 may be zero. In order to apply the abstract error representation formula of Lemma 3.1, we introduce the Lagrangians $\mathcal{L}\colon X \times$

$X \to \mathbb{R}$, $\widetilde{\mathcal{L}} : X_k^r \times X_k^r \to \mathbb{R}$, and $\widetilde{\mathcal{L}}_h : X_{kh}^{r,s} \times X_{kh}^{r,s} \to \mathbb{R}$ by

$$\mathcal{L}(\boldsymbol{u}, \boldsymbol{z}) \coloneqq J(\boldsymbol{u}) + \int_I \left\{ (\boldsymbol{f} - \partial_t \boldsymbol{v}, \boldsymbol{w}) - a(\boldsymbol{u})(\boldsymbol{z}) \right\} \mathrm{d}t - (\boldsymbol{v}(0) - \boldsymbol{v}^{\boldsymbol{0}}, \boldsymbol{w}(0)),$$

$$\begin{aligned}\widetilde{\mathcal{L}}(\boldsymbol{u_k}, \boldsymbol{z_k}) \coloneqq J(\boldsymbol{u_k}) &+ \sum_{m=1}^{M} \int_{I_m} \left\{ (\boldsymbol{f} - \partial_t \boldsymbol{v_k}, \boldsymbol{w_k}) - a(\boldsymbol{u_k})(\boldsymbol{z_k}) \right\} \mathrm{d}t \\ &- \sum_{m=0}^{M-1} ([\boldsymbol{v_k}]_m, \boldsymbol{w}_{\boldsymbol{k},m}^+) - (\boldsymbol{v}_{\boldsymbol{k},0}^- - \boldsymbol{v}^{\boldsymbol{0}}, \boldsymbol{w}_{\boldsymbol{k},0}^-),\end{aligned}$$

$$\widetilde{\mathcal{L}}_h(\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}}) \coloneqq \widetilde{\mathcal{L}}(\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}}) - \mathcal{S}_h(\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}})$$

with

$$\mathcal{S}_h(\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}}) \coloneqq \sum_{m=1}^{M} \int_{I_m} s_h^m(\boldsymbol{u_{kh}})(\boldsymbol{z_{kh}}) \, \mathrm{d}t.$$

Remark 3.1. The Lagrange multipliers $\boldsymbol{z} = (\boldsymbol{w}, q)^T$, $\boldsymbol{z_k} = (\boldsymbol{w_k}, q_k)^T$, and $\boldsymbol{z_{kh}} = (\boldsymbol{w_{kh}}, q_{kh})^T$ introduced in this context, are usually called *dual variables* in contrast to the *primal variables* $\boldsymbol{u} = (\boldsymbol{v}, p)^T$, $\boldsymbol{u_k} = (\boldsymbol{v_k}, p_k)^T$, and $\boldsymbol{u_{kh}} = (\boldsymbol{v_{kh}}, p_{kh})^T$.

Using the Lagrangians, we can express the functional values of the continuous, semi-discrete, and fully discrete solution as follows:

$$J(\boldsymbol{u}) = \mathcal{L}(\boldsymbol{u}, \boldsymbol{\varphi}) \qquad \forall \boldsymbol{\varphi} \in X, \tag{10a}$$

$$J(\boldsymbol{u_k}) = \widetilde{\mathcal{L}}(\boldsymbol{u_k}, \boldsymbol{\varphi}) \qquad \forall \boldsymbol{\varphi} \in X_k^r, \tag{10b}$$

$$J(\boldsymbol{u_{kh}}) = \widetilde{\mathcal{L}}_h(\boldsymbol{u_{kh}}, \boldsymbol{\varphi}) \quad \forall \boldsymbol{\varphi} \in X_{kh}^{r,s}. \tag{10c}$$

Since we want to separate the influences of the temporal and spatial discretization, we split the total discretization error as

$$J(\boldsymbol{u}) - J(\boldsymbol{u_{kh}}) = (J(\boldsymbol{u}) - J(\boldsymbol{u_k})) + (J(\boldsymbol{u_k}) - J(\boldsymbol{u_{kh}})),$$

where $\boldsymbol{u}$ denotes the continuous solution, $\boldsymbol{u_k}$ the semi-discrete solution of the dG(r) discretization in time, and $\boldsymbol{u_{kh}}$ the fully discrete solution of the cG(s)dG(r) discretization. Note that these solutions are given as the first

component of stationary points of the corresponding Lagrangians, since

$$\begin{aligned}\mathcal{L}'_{\boldsymbol{z}}(\boldsymbol{u},\boldsymbol{z})(\boldsymbol{\varphi}) &= 0 \quad \forall \boldsymbol{\varphi} \in X,\\ \widetilde{\mathcal{L}}'_{\boldsymbol{z}}(\boldsymbol{u_k},\boldsymbol{z_k})(\boldsymbol{\varphi}) &= 0 \quad \forall \boldsymbol{\varphi} \in X_k^r,\\ \widetilde{\mathcal{L}}'_{h,\boldsymbol{z}}(\boldsymbol{u_{kh}},\boldsymbol{z_{kh}})(\boldsymbol{\varphi}) &= 0 \quad \forall \boldsymbol{\varphi} \in X_{kh}^{r,s}\end{aligned}$$

are just the equations for the continuous, semi-discrete and fully discrete problem.

We are now able to state the following theorem:

Theorem 3.2. *Let $(\boldsymbol{u},\boldsymbol{z})^T$, $(\boldsymbol{u_k},\boldsymbol{z_k})^T$, and $(\boldsymbol{u_{kh}},\boldsymbol{z_{kh}})^T$ denote stationary points of $\mathcal{L}$, $\widetilde{\mathcal{L}}$, and $\widetilde{\mathcal{L}}_h$ on different discretization levels, i. e.,*

$$\begin{aligned}\mathcal{L}'(\boldsymbol{u},\boldsymbol{z})(\boldsymbol{\delta u},\boldsymbol{\delta z}) = \widetilde{\mathcal{L}}'(\boldsymbol{u},\boldsymbol{z})(\boldsymbol{\delta u},\boldsymbol{\delta z}) &= 0 \quad \forall (\boldsymbol{\delta u},\boldsymbol{\delta z})^T \in X \times X,\\ \widetilde{\mathcal{L}}'(\boldsymbol{u_k},\boldsymbol{z_k})(\boldsymbol{\delta u_k},\boldsymbol{\delta z_k}) &= 0 \quad \forall (\boldsymbol{\delta u_k},\boldsymbol{\delta z_k})^T \in X_k^r \times X_k^r,\\ \widetilde{\mathcal{L}}'_h(\boldsymbol{u_{kh}},\boldsymbol{z_{kh}})(\boldsymbol{\delta u_{kh}},\boldsymbol{\delta z_{kh}}) &= 0 \quad \forall (\boldsymbol{\delta u_{kh}},\boldsymbol{\delta z_{kh}})^T \in X_{kh}^{r,s} \times X_{kh}^{r,s}.\end{aligned}$$

Then, there hold the following error representation formulas for the discretization errors in time and space:

$$\begin{aligned}J(\boldsymbol{u}) - J(\boldsymbol{u_k}) &= \frac{1}{2}\widetilde{\mathcal{L}}'(\boldsymbol{u_k},\boldsymbol{z_k})(\boldsymbol{u}-\tilde{\boldsymbol{u}}_{\boldsymbol{k}},\boldsymbol{z}-\tilde{\boldsymbol{z}}_{\boldsymbol{k}}) + \mathcal{R}_k,\\ J(\boldsymbol{u_k}) - J(\boldsymbol{u_{kh}}) &= \frac{1}{2}\widetilde{\mathcal{L}}'(\boldsymbol{u_{kh}},\boldsymbol{z_{kh}})(\boldsymbol{u_k}-\tilde{\boldsymbol{u}}_{\boldsymbol{kh}},\boldsymbol{z_k}-\tilde{\boldsymbol{z}}_{\boldsymbol{kh}})\\ &\quad + \frac{1}{2}\mathcal{S}'_h(\boldsymbol{u_{kh}},\boldsymbol{z_{kh}})(\tilde{\boldsymbol{u}}_{\boldsymbol{kh}}-\boldsymbol{u_{kh}},\tilde{\boldsymbol{z}}_{\boldsymbol{kh}}-\boldsymbol{z_{kh}}) + \mathcal{S}_h(\boldsymbol{u_{kh}},\boldsymbol{z_{kh}})\\ &\quad + \mathcal{R}_h.\end{aligned}$$

Here, $(\tilde{\boldsymbol{u}}_{\boldsymbol{k}},\tilde{\boldsymbol{z}}_{\boldsymbol{k}})^T \in X_k^r \times X_k^r$ and $(\tilde{\boldsymbol{u}}_{\boldsymbol{kh}},\tilde{\boldsymbol{z}}_{\boldsymbol{kh}})^T \in X_{kh}^{r,s} \times X_{kh}^{r,s}$ can be chosen arbitrarily and the remainder terms $\mathcal{R}_k$ and $\mathcal{R}_h$ have the same structure as in Lemma 3.1.

Proof. Due to (10), we may especially write

$$J(\boldsymbol{u}) - J(\boldsymbol{u_k}) = \mathcal{L}(\boldsymbol{u},\boldsymbol{z}) - \widetilde{\mathcal{L}}(\boldsymbol{u_k},\boldsymbol{z_k}) = \widetilde{\mathcal{L}}(\boldsymbol{u},\boldsymbol{z}) - \widetilde{\mathcal{L}}(\boldsymbol{u_k},\boldsymbol{z_k}), \tag{11a}$$

$$J(\boldsymbol{u_k}) - J(\boldsymbol{u_{kh}}) = \widetilde{\mathcal{L}}(\boldsymbol{u_k},\boldsymbol{z_k}) - \widetilde{\mathcal{L}}_h(\boldsymbol{u_{kh}},\boldsymbol{z_{kh}}). \tag{11b}$$

Here, we have used the fact that

$$J(\boldsymbol{u}) = \mathcal{L}(\boldsymbol{u},\boldsymbol{z}) = \widetilde{\mathcal{L}}(\boldsymbol{u},\boldsymbol{z}),$$

since the first component $\boldsymbol{v}$ of $\boldsymbol{u} \in X$ is continuous and hence the additional jump terms in $\widetilde{\mathcal{L}}$ compared to $\mathcal{L}$ vanish. Next, we apply Lemma 3.1 with

$$\begin{aligned} &L = \widetilde{\mathcal{L}}, \quad \widetilde{L} = \widetilde{\mathcal{L}}, \quad Y_1 = X \times X, \qquad Y_2 = X_k^r \times X_k^r \qquad \text{for (11a)}, \\ &L = \widetilde{\mathcal{L}}, \quad \widetilde{L} = \widetilde{\mathcal{L}}_h, \quad Y_1 = X_k^r \times X_k^r, \quad Y_2 = X_{kh}^{r,s} \times X_{kh}^{r,s} \quad \text{for (11b)}. \end{aligned}$$

In the second case, we have $Y_2 \subseteq Y_1$ since $X_{kh}^{r,s} \subseteq X_k^r$. Hence, we can take $Y \coloneqq Y_1$ and condition (8) is fulfilled automatically.

For the first case, he have to choose $Y \coloneqq Y_1 + Y_2$ since $X_k^r \not\subseteq X$. Thus, we must check condition (8) which reads

$$\widetilde{\mathcal{L}}'(\boldsymbol{u}, \boldsymbol{z})(\boldsymbol{u_k}, \boldsymbol{z_k}) = 0$$

or equivalently

$$\widetilde{\mathcal{L}}'_{\boldsymbol{u}}(\boldsymbol{u}, \boldsymbol{z})(\boldsymbol{u_k}) = 0 \quad \text{and} \quad \widetilde{\mathcal{L}}'_{\boldsymbol{z}}(\boldsymbol{u}, \boldsymbol{z})(\boldsymbol{z_k}) = 0.$$

We only show the proof of the second condition

$$\widetilde{\mathcal{L}}'_{\boldsymbol{z}}(\boldsymbol{u}, \boldsymbol{z})(\boldsymbol{z_k}) = 0. \tag{12}$$

The first one can be handled analogously. Due to the continuity of the first component of the continuous solution $\boldsymbol{u}$ with respect to time, the jump terms and the initial condition in $\widetilde{\mathcal{L}}$ vanish on $\boldsymbol{u} \in X$. Hence, equation (12) may be rewritten as

$$\sum_{m=1}^{M} \int_{I_m} \left\{ (\boldsymbol{f} - \partial_t \boldsymbol{v}, \boldsymbol{w_k}) - a(\boldsymbol{u})(\boldsymbol{z_k}) \right\} \mathrm{d}t = 0.$$

By construction, the continuous solution $\boldsymbol{u}$ fulfills

$$\int_I \left\{ (\partial_t \boldsymbol{v}, \boldsymbol{\psi}) + a(\boldsymbol{u})(\boldsymbol{\varphi}) \right\} \mathrm{d}t = \int_I (\boldsymbol{f}, \boldsymbol{\psi}) \, \mathrm{d}t \quad \forall \boldsymbol{\varphi} = (\boldsymbol{\psi}, \chi)^T \in X. \tag{13}$$

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 $\boldsymbol{\psi}$ in (13), this equation also holds true for all $\boldsymbol{\varphi} \in L^2(I, H_0^1(\Omega)^d \times L^2(\Omega)/\mathbb{R})$. The inclusion $\boldsymbol{z_k} \in X_k^r \subseteq L^2(I, H_0^1(\Omega)^d \times L^2(\Omega)/\mathbb{R})$ then implies that condition (12) is fulfilled.

Finally, the assertion of the theorem is a direct consequence of Lemma 3.1 applied to the separated errors (11). □

Introducing the primal and dual residual

$$\begin{aligned}\rho(\boldsymbol{u})(\boldsymbol{\varphi}) &\coloneqq \widetilde{\mathcal{L}}'_{\boldsymbol{z}}(\boldsymbol{u},\boldsymbol{z})(\boldsymbol{\varphi}),\\ \rho^*(\boldsymbol{u},\boldsymbol{z})(\boldsymbol{\varphi}) &\coloneqq \widetilde{\mathcal{L}}'_{\boldsymbol{u}}(\boldsymbol{u},\boldsymbol{z})(\boldsymbol{\varphi}),\end{aligned}$$

the result of Theorem 3.2 may be rewritten as

$$J(\boldsymbol{u}) - J(\boldsymbol{u_k}) \approx \frac{1}{2}\big\{\rho(\boldsymbol{u_k})(\boldsymbol{z}-\tilde{\boldsymbol{z}}_{\boldsymbol{k}}) + \rho^*(\boldsymbol{u_k},\boldsymbol{z_k})(\boldsymbol{u}-\tilde{\boldsymbol{u}}_{\boldsymbol{k}})\big\}, \tag{14a}$$

$$J(\boldsymbol{u_k}) - J(\boldsymbol{u_{kh}}) \approx \frac{1}{2}\big\{\rho(\boldsymbol{u_{kh}})(\boldsymbol{z_k}-\tilde{\boldsymbol{z}}_{\boldsymbol{kh}}) + \rho^*(\boldsymbol{u_{kh}},\boldsymbol{z_{kh}})(\boldsymbol{u_k}-\tilde{\boldsymbol{u}}_{\boldsymbol{kh}})\big\}, \tag{14b}$$

where we have neglected the remainder terms $\mathcal{R}_k$ and $\mathcal{R}_h$ as well as the additional terms due to stabilization which can be assumed to be small because they contain small stabilization parameters. At least, numerical results show that they are indeed negligible, see Section 5.

4. Numerical realization

In this section, 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 $\boldsymbol{z} \in X$, $\boldsymbol{z_k} \in X_k^r$, and $\boldsymbol{z_{kh}} \in X_{kh}^{r,s}$. They are given as solutions of

$$\begin{aligned}\mathcal{L}'_{\boldsymbol{u}}(\boldsymbol{u},\boldsymbol{z})(\boldsymbol{\varphi}) &= 0 \quad \forall \boldsymbol{\varphi} \in X,\\ \widetilde{\mathcal{L}}'_{\boldsymbol{u}}(\boldsymbol{u_k},\boldsymbol{z_k})(\boldsymbol{\varphi}) &= 0 \quad \forall \boldsymbol{\varphi} \in X_k^r,\\ \widetilde{\mathcal{L}}'_{h,\boldsymbol{u}}(\boldsymbol{u_{kh}},\boldsymbol{z_{kh}})(\boldsymbol{\varphi}) &= 0 \quad \forall \boldsymbol{\varphi} \in X_{kh}^{r,s}.\end{aligned}$$

We want to show the precise form of these derivatives, i. e., the equations the dual solutions have to fulfill. The continuous dual solution $\boldsymbol{z} = (\boldsymbol{w}, q)^T \in X$ is the solution of

$$\begin{aligned}&\int_I \big\{(\boldsymbol{\psi}, -\partial_t \boldsymbol{w}) + a'(\boldsymbol{u})(\boldsymbol{\varphi},\boldsymbol{z})\big\}\,\mathrm{d}t + (\boldsymbol{\psi}(T), \boldsymbol{w}(T))\\ &\qquad = \int_I J_1'(\boldsymbol{u})(\boldsymbol{\varphi})\,\mathrm{d}t + J_2'(\boldsymbol{u}(T))(\boldsymbol{\varphi}(T)) \quad \forall \boldsymbol{\varphi} = (\boldsymbol{\psi},\chi)^T \in X,\end{aligned} \tag{15}$$

where we have integrated by parts which is admissible for functions in X, see, for instance, [24]. The semi-discrete dual solution $\boldsymbol{z_k} = (\boldsymbol{w_k}, q_k)^T \in X_k^r$ and the fully discrete dual solution $\boldsymbol{z_{kh}} = (\boldsymbol{w_{kh}}, q_{kh})^T \in X_{kh}^{r,s}$ fulfill

$$\sum_{m=1}^{M} \int_{I_m} \left\{ (\boldsymbol{\psi}, -\partial_t \boldsymbol{w_k}) + a'(\boldsymbol{u_k})(\boldsymbol{\varphi}, \boldsymbol{z_k}) \right\} \, \mathrm{d}t - \sum_{m=0}^{M-1} (\boldsymbol{\psi}_m^-, [\boldsymbol{w_k}]_m) + (\boldsymbol{\psi}_M^-, \boldsymbol{w}_{\boldsymbol{k},M}^-)$$

$$= \int_I J_1'(\boldsymbol{u_k})(\boldsymbol{\varphi}) \, \mathrm{d}t + J_2'(\boldsymbol{u}_{\boldsymbol{k},M}^-)(\boldsymbol{\varphi}_M^-) \quad \forall \boldsymbol{\varphi} = (\boldsymbol{\psi}, \chi)^T \in X_k^r \quad (16)$$

and

$$\sum_{m=1}^{M} \int_{I_m} \left\{ (\boldsymbol{\psi}, -\partial_t \boldsymbol{w_{kh}}) + a(\boldsymbol{u_{kh}})(\boldsymbol{\varphi}, \boldsymbol{z_{kh}}) + s_h^{m\prime}(\boldsymbol{u_{kh}})(\boldsymbol{\varphi}, \boldsymbol{z_{kh}}) \right\} \, \mathrm{d}t$$

$$- \sum_{m=0}^{M-1} (\boldsymbol{\psi}_m^-, [\boldsymbol{w_{kh}}]_m) + (\boldsymbol{\psi}_M^-, \boldsymbol{w}_{\boldsymbol{kh},M}^-)$$

$$= \int_I J_1'(\boldsymbol{u_{kh}})(\boldsymbol{\varphi}) \, \mathrm{d}t + J_2'(\boldsymbol{u}_{\boldsymbol{kh},M}^-)(\boldsymbol{\varphi}_M^-) \quad \forall \boldsymbol{\varphi} = (\boldsymbol{\psi}, \chi)^T \in X_{kh}^{r,s}, \quad (17)$$

respectively.

Note that for solving the dual problems (17), the primal solution $\boldsymbol{u_{kh}}$ is needed on the whole time interval $\bar{I}$ due to the nonlinear structure 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, for instance, in [25], [26] or [27]. However, since in the last years the capacities of main memory and hard disk drives have 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 spatial 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, we also refer to [28].

4.1. Numerical evaluation of the error estimator

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 $\tilde{\boldsymbol{u}}_{\boldsymbol{k}}$, $\tilde{\boldsymbol{z}}_{\boldsymbol{k}}$, $\tilde{\boldsymbol{u}}_{\boldsymbol{kh}}$, and $\tilde{\boldsymbol{z}}_{\boldsymbol{kh}}$ can be chosen arbitrarily in the corresponding spaces, the so-called *weights*, i. e., $\boldsymbol{u} - \tilde{\boldsymbol{u}}_{\boldsymbol{k}}$, $\boldsymbol{z} - \tilde{\boldsymbol{z}}_{\boldsymbol{k}}$, and so on, are mainly interpolation errors. We approximate these interpolation errors by higher order reconstructions of 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 [23] for more details on this topic and alternative approaches.

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

$$\begin{aligned}
\boldsymbol{v} - \tilde{\boldsymbol{v}}_{\boldsymbol{k}} &\approx \boldsymbol{\Pi}_{\boldsymbol{k}}^{(\boldsymbol{v})} \boldsymbol{v}_{\boldsymbol{k}}, & \boldsymbol{v}_{\boldsymbol{k}} - \tilde{\boldsymbol{v}}_{\boldsymbol{kh}} &\approx \boldsymbol{\Pi}_{\boldsymbol{h}}^{(\boldsymbol{v})} \boldsymbol{v}_{\boldsymbol{kh}}, \\
p - \tilde{p}_k &\approx \Pi_k^{(p)} p_k, & p_k - \tilde{p}_{kh} &\approx \Pi_h^{(p)} p_{kh}, \\
\boldsymbol{w} - \tilde{\boldsymbol{w}}_{\boldsymbol{k}} &\approx \boldsymbol{\Pi}_{\boldsymbol{k}}^{(\boldsymbol{v})} \boldsymbol{w}_{\boldsymbol{k}}, & \boldsymbol{w}_{\boldsymbol{k}} - \tilde{\boldsymbol{w}}_{\boldsymbol{kh}} &\approx \boldsymbol{\Pi}_{\boldsymbol{h}}^{(\boldsymbol{v})} \boldsymbol{w}_{\boldsymbol{kh}}, \\
q - \tilde{q}_k &\approx \Pi_k^{(p)} q_k, & q_k - \tilde{q}_{kh} &\approx \Pi_h^{(p)} q_{kh}.
\end{aligned}$$

The operators $\boldsymbol{\Pi}_{\boldsymbol{k}}^{(\boldsymbol{v})}$, $\Pi_k^{(p)}$ as well as $\boldsymbol{\Pi}_{\boldsymbol{h}}^{(\boldsymbol{v})}$ and $\Pi_h^{(p)}$ are chosen as

cG(s)dG(0):

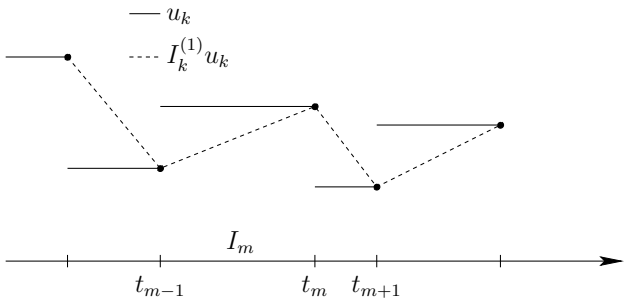
$$\begin{aligned}
\boldsymbol{\Pi}_{\boldsymbol{k}}^{(\boldsymbol{v})} &\coloneqq \boldsymbol{I}_{\boldsymbol{k}}^{(\boldsymbol{1})} - \mathbf{id}, & \boldsymbol{\Pi}_{\boldsymbol{h}}^{(\boldsymbol{v})} &\coloneqq \boldsymbol{I}_{\boldsymbol{2h}}^{(\boldsymbol{2s})} - \mathbf{id}, \\
\Pi_k^{(p)} &\coloneqq I_k^{(1)} - \mathrm{id}, & \Pi_h^{(p)} &\coloneqq I_{2h}^{(2s)} - \mathrm{id},
\end{aligned}$$

where $I_k^{(1)}$ is given as in Figure 3(a) and $\boldsymbol{I}_{\boldsymbol{k}}^{(\boldsymbol{1})}$ acts component-wise as $I_k^{(1)}$.

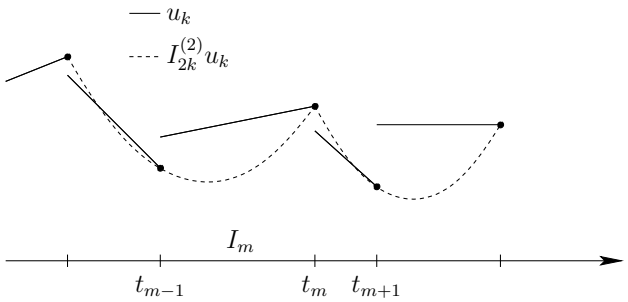
cG(s)dG(1):

$$\begin{aligned}
\boldsymbol{\Pi}_{\boldsymbol{k}}^{(\boldsymbol{v})} &\coloneqq \boldsymbol{I}_{\boldsymbol{2k}}^{(\boldsymbol{2})} - \mathbf{id}, & \boldsymbol{\Pi}_{\boldsymbol{h}}^{(\boldsymbol{v})} &\coloneqq \boldsymbol{I}_{\boldsymbol{2h}}^{(\boldsymbol{2s})} - \mathbf{id}, \\
\Pi_k^{(p)} &\coloneqq I_{2k}^{(2)} - \mathrm{id}, & \Pi_h^{(p)} &\coloneqq I_{2h}^{(2s)} - \mathrm{id},
\end{aligned}$$

where $I_{2k}^{(2)}$ is given as in Figure 3(b) and $\boldsymbol{I}_{\boldsymbol{2k}}^{(\boldsymbol{2})}$ acts component-wise as $I_{2k}^{(2)}$.



(a) Continuous piecewise linear interpolation of a discontinuous piecewise constant function



(b) Continuous piecewise quadratic interpolation of a discontinuous piecewise linear function

Figure 3: Interpolation operators $I_k^{(1)}$ and $I_{2k}^{(2)}$

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

$$(I_{2h}^{(2s)} u_{kh})(t) \coloneqq I_{2h}^{(2s)}(u_{kh}(t)).$$

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 fully discrete versions, i. e., we replace

$$\rho(\boldsymbol{u_k})(\boldsymbol{z} - \tilde{\boldsymbol{z}}_{\boldsymbol{k}}) \text{ by } \rho(\boldsymbol{u_{kh}})(\boldsymbol{\Pi_k z_{kh}})$$

and

$$\rho^*(\boldsymbol{u_k}, \boldsymbol{z_k})(\boldsymbol{u} - \tilde{\boldsymbol{u}}_{\boldsymbol{k}}) \text{ by } \rho^*(\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}})(\boldsymbol{\Pi_k u_{kh}})$$

with $\boldsymbol{\Pi_k z_{kh}} \coloneqq (\boldsymbol{\Pi}_{\boldsymbol{k}}^{(\boldsymbol{v})} \boldsymbol{w_{kh}}, \Pi_k^{(p)} q_{kh})^T$ and $\boldsymbol{\Pi_k u_{kh}} \coloneqq (\boldsymbol{\Pi}_{\boldsymbol{k}}^{(\boldsymbol{v})} \boldsymbol{v_{kh}}, \Pi_k^{(p)} p_{kh})^T$. The replacement in the weights is well accepted while the replacement in the residuals, i. e., the replacement of the linearization point, might seem critical. One could think of replacing the unknown solutions also by higher order interpolations as we did in the weights. However, in our numerical examples we see that this additional effort is not necessary to obtain quantitatively good results.

Remark 4.1. This observation is also substantiated by the fact that the replacement of the linearization point introduces an additional error which usually is of higher order. This can be seen as follows: The introduced error can be expressed as

$$\widetilde{\mathcal{L}}'(\boldsymbol{\zeta_k})(\boldsymbol{\zeta}-\tilde{\boldsymbol{\zeta}}_{\boldsymbol{k}}) - \widetilde{\mathcal{L}}'(\boldsymbol{\zeta_{kh}})(\boldsymbol{\zeta}-\tilde{\boldsymbol{\zeta}}_{\boldsymbol{k}}) = \int_0^1 \widetilde{\mathcal{L}}''(\boldsymbol{\zeta_{kh}} + s(\boldsymbol{\zeta_k} - \boldsymbol{\zeta_{kh}}))(\boldsymbol{\zeta_k} - \boldsymbol{\zeta_{kh}}, \boldsymbol{\zeta} - \tilde{\boldsymbol{\zeta}}_{\boldsymbol{k}}) \, \mathrm{d}s$$

with $\boldsymbol{\zeta} = (\boldsymbol{u}, \boldsymbol{z})^T$, $\boldsymbol{\zeta_k} = (\boldsymbol{u_k}, \boldsymbol{z_k})^T$, and $\boldsymbol{\zeta_{kh}} = (\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}})^T$. By choosing an appropriate interpolant for $\tilde{\boldsymbol{\zeta}}_{\boldsymbol{k}}$, this identity shows that the discussed replacement introduces an error of the order $O(h^2 k)$ 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, we refer to [28] or [7].

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

$$J(\boldsymbol{u}) - J(\boldsymbol{u_{kh}}) \approx \eta_k + \eta_h$$

with

$$\eta_k \coloneqq \frac{1}{2}\big\{\rho(\boldsymbol{u_{kh}})(\boldsymbol{\Pi_k z_{kh}}) + \rho^*(\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}})(\boldsymbol{\Pi_k u_{kh}})\big\},$$
$$\eta_h \coloneqq \frac{1}{2}\big\{\rho(\boldsymbol{u_{kh}})(\boldsymbol{\Pi_h z_{kh}}) + \rho^*(\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}})(\boldsymbol{\Pi_h u_{kh}})\big\}.$$

To get an impression of what terms have to be evaluated, we present exemplarily for the backward Euler variant of the cG(s)dG(0) discretization their precise form. To this end, temporal integrals involving $\boldsymbol{u_{kh}}$ and $\boldsymbol{z_{kh}}$ are approximated by the box rule whereas those involving $\boldsymbol{I_k^{(1)}u_{kh}}$ and $\boldsymbol{I_k^{(1)}u_{kh}}$ are evaluated using the trapezoidal rule. This leads to the following representation:

$$\begin{aligned}
\rho(\boldsymbol{u_{kh}})(\boldsymbol{\Pi_k z_{kh}}) = \sum_{m=1}^{M}\Big\{&(\boldsymbol{V_m} - \boldsymbol{V_{m-1}}, \boldsymbol{W_m} - \boldsymbol{W_{m-1}})\\
&+ \frac{k_m}{2}a(\boldsymbol{U_m})(\boldsymbol{Z_m} - \boldsymbol{Z_{m-1}})\\
&+ \frac{k_m}{2}(\boldsymbol{f}(t_{m-1}), \boldsymbol{W_{m-1}}) - \frac{k_m}{2}(\boldsymbol{f}(t_m), \boldsymbol{W_m})\Big\},\\
\rho^*(\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}})(\boldsymbol{\Pi_k u_{kh}}) = \sum_{m=1}^{M}\Big\{&\frac{k_m}{2}a'(\boldsymbol{U_m})(\boldsymbol{U_m} - \boldsymbol{U_{m-1}}, \boldsymbol{Z_m})\\
&- \frac{k_m}{2}J_1'(\boldsymbol{U_m})(\boldsymbol{U_m} - \boldsymbol{U_{m-1}})\Big\},\\
\rho(\boldsymbol{u_{kh}})(\boldsymbol{\Pi_h z_{kh}}) = \sum_{m=1}^{M}\big\{&k_m(\boldsymbol{f}(t_m), \boldsymbol{I_{2h}^{(2s)}W_m} - \boldsymbol{W_m})\\
&- k_m a(\boldsymbol{U_m})(\boldsymbol{I_{2h}^{(2s)}Z_m} - \boldsymbol{Z_m})\\
&- (\boldsymbol{V_m} - \boldsymbol{V_{m-1}}, \boldsymbol{I_{2h}^{(2s)}W_m} - \boldsymbol{W_m})\big\}\\
&- (\boldsymbol{V_0} - \boldsymbol{v^0}, \boldsymbol{I_{2h}^{(2s)}W_0} - \boldsymbol{W_0}),
\end{aligned}$$

$$\begin{aligned}\rho^*(\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}})(\boldsymbol{\Pi_h u_{kh}}) = \sum_{m=1}^{M} \big\{ & k_m J_1'(\boldsymbol{U_m})(\boldsymbol{I_{2h}^{(2s)}} \boldsymbol{U_m} - \boldsymbol{U_m}) \\ & - k_m a'(\boldsymbol{U_m})(\boldsymbol{I_{2h}^{(2s)}} \boldsymbol{U_m} - \boldsymbol{U_m}, \boldsymbol{Z_m}) \\ & + (\boldsymbol{I_{2h}^{(2s)}} \boldsymbol{V_{m-1}} - \boldsymbol{V_{m-1}}, \boldsymbol{W_m} - \boldsymbol{W_{m-1}}) \big\} \\ & + J_2'(\boldsymbol{U_M})(\boldsymbol{I_{2h}^{(2s)}} \boldsymbol{U_M} - \boldsymbol{U_M}) \\ & - (\boldsymbol{I_{2h}^{(2s)}} \boldsymbol{V_M} - \boldsymbol{V_M}, \boldsymbol{W_M}).\end{aligned}$$

Similar expressions are obtained for the cG(s)dG(1) discretization. Of course, quadrature rules of higher order have to be applied in order to exactly evaluate the temporal integrals.

The a posteriori error estimators developed above serve for two purposes: Firstly, the quantitative assessment of the discretization error and secondly the adaptive refinement 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 cell-wise or node-wise contributions. These quantities are then called *local error indicators*.

To this end, we split the overall error estimators into their contributions on each subinterval I_m by

$$\eta_k = \sum_{m=1}^{M} \eta_k^m \quad \text{and} \quad \eta_h = \sum_{m=0}^{M} \eta_h^m,$$

where the interval-wise error estimators are defined analogously to the global error estimators by

$$\begin{aligned}\eta_k^m &\coloneqq \frac{1}{2}\big\{\rho_m(\boldsymbol{u_k})(\boldsymbol{z} - \boldsymbol{\tilde{z}_k}) + \rho_m^*(\boldsymbol{u_k}, \boldsymbol{z_k})(\boldsymbol{u} - \boldsymbol{\tilde{u}_k})\big\}, \\ \eta_h^m &\coloneqq \frac{1}{2}\big\{\rho_m(\boldsymbol{u_{kh}})(\boldsymbol{z_k} - \boldsymbol{\tilde{z}_{kh}}) + \rho_m^*(\boldsymbol{u_{kh}}, \boldsymbol{z_{kh}})(\boldsymbol{u_k} - \boldsymbol{\tilde{u}_{kh}})\big\},\end{aligned}$$

but involve only those parts ρ_m and ρ_m^* of the global residuals ρ and ρ^* belonging to the subinterval I_m or to the initial time $t = 0$ for $m = 0$.

While the hereby obtained local contributions η_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:

$$\eta_h^m = \sum_{K \in \mathcal{T}_h^m} \eta_{h,K}^m, \quad m = 0, \ldots, M.$$

However, a simple splitting into cell-wise contributions leads to a large overestimation of the actual error due to oscillatory behavior of the residuals, see [29]. A commonly used way to overcome this difficulty is to apply cell-wise integration by parts in space, see, e. g., [30, 23]. The resulting local error indicators involve the strong residual of the equation as well as jumps of the discrete solution over faces of cells.

Another way of overcoming this problem which we applied in our computations and which avoids having to evaluate strong residuals and jumps over faces of cells was presented in [31]. For details on the application of this approach to nonstationary problems, we refer to [7] and [6].

For setting up an efficient adaptive algorithm, it is essential that the temporal error estimator η_k is independent of the refinement of the spatial discretization and vice versa. We will see in Section 5, that this is (almost) the case for both η_k and η_h. However, the local contributions η_h^m depend linearly on the local size k_m of the subintervals 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 $\hat{\eta}_{h,K}^m$ independent of k_m and hence usable in a simultaneous mesh adaption algorithm as presented below by rescaling:

$$\hat{\eta}_{h,K}^m := \frac{T}{k_m} \eta_{h,K}^m, \quad K \in \mathcal{T}_h^m, \quad m = 0, \ldots, M.$$

This scaling has the following special property: If the rescaled spatial error indicators $\hat{\eta}_h^m$ fulfill

$$\hat{\eta}_h^m = \sum_{K \in \mathcal{T}_h^m} \hat{\eta}_{h,K}^m < \mathrm{TOL},$$

we then have for the whole spatial error estimator

$$\eta_h = \sum_{m=0}^{M} \eta_h^m = \sum_{m=0}^{M} \frac{k_m}{T} \hat{\eta}_h^m < \frac{\mathrm{TOL}}{T} \sum_{m=0}^{M} k_m = \mathrm{TOL}.$$

Remark 4.2. If the cells to be refined within an adaptive mesh refinement procedure are not chosen by a tolerance-based selection criterion, the simpler rescaling

$$\hat{\eta}_{h,K}^m \coloneqq \frac{1}{k_m} \eta_{h,K}^m$$

is sufficient.

This supplies us with two sets of error indicators which will be used within the adaptive algorithm presented in the next subsection for an efficient automatic adaptation of the temporal and spatial discretizations. These sets are given by

$$\Sigma_k \coloneqq \{ \eta_k^m \mid m = 1, \dots, M \} \text{ and } \Sigma_h \coloneqq \left\{ \hat{\eta}_{h,K}^m \;\middle|\; K \in \mathcal{T}_h^m,\ m = 0, \dots, M \right\}.$$

Remark 4.3. Note that for efficiency reasons it is necessary to treat the cell-wise error indicators of all spatial meshes simultaneously rather than for each mesh separately. If we used $M + 1$ different sets of cell-wise error indicators

$$\Sigma_h^m \coloneqq \left\{ \hat{\eta}_{h,K}^m \;\middle|\; K \in \mathcal{T}_h^m \right\}, \quad m = 0, \dots, M,$$

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. The other way around, 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 Σ_h does not produce such inefficient meshes because the error indicators are sorted "globally".

4.2. Adaptive algorithm

In this subsection, we present an adaptive refinement algorithm which uses the developed a posteriori error estimators of Section 3 to automatically adjust 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. This requires a precise quantitative assessment of both discretization errors as it is available with the derived error estimators (see Section 5 for numerical results).

If the functional value $J(\boldsymbol{u})$ is to be computed to a given accuracy TOL, this can be achieved by refining each discretization as long as the corresponding part of the error is greater than $\frac{\text{TOL}}{2}$. 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 sequel, we present an adaptive algorithm which balances the initial temporal and spatial discretization errors and keeps them balanced during further refinement without having to prescribe a certain accuracy TOL. This leads to an algorithm which uses the given computational resources efficiently in order to achieve an accuracy as good as possible. The stopping criterion therefore is based on reaching a prescribed maximum number of degrees of freedom (determined by the given architecture) which must not be exceeded rather than on reaching the desired accuracy TOL. Of course, these stopping criterions can easily be exchanged by instead checking if

$$|\eta| = |\eta_k + \eta_h| \leq \text{TOL},$$

at least under consideration of the problem mentioned above.

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 propose to proceed as in Algorithm 1.

Remark 4.4. The behavior of Algorithm 1 strongly depends on the choice of κ. Choosing κ 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 actually are of the same size. On the other hand, choosing κ too large, makes the algorithm inefficient because both discretizations are refined although the total discretization error is dominated by only the temporal or the spatial discretization error. Numerical tests show $\kappa \approx 3$ to be a good choice.

When refining a discretization, the cells (or time intervals) which are to be refined are chosen using sets Σ_k or Σ_h of error indicators like the ones

Algorithm 1 Adaptive refinement algorithm

Choose an initial temporal and spatial discretization $\mathcal{T}_{k_0,h_0}$.
Set $n = 0$.
loop
  Compute the primal and dual solution $\boldsymbol{u_{k_n h_n}}$ and $\boldsymbol{z_{k_n h_n}}$.
  Evaluate the a posteriori error estimators η_{k_n} and η_{h_n}.
  if the maximum number of degrees of freedom is reached **then**
    return
  if $|\eta_{k_n}| > \kappa\,|\eta_{h_n}|$ **then**
    Adapt the temporal discretization.
  else if $|\eta_{h_n}| > \kappa\,|\eta_{k_n}|$ **then**
    Adapt the spatial discretization.
  else
    Adapt the temporal and spatial discretization.
  Increase n.

shown at the end of the previous section. Thus, we have to select subsets $\Sigma_k^R \subseteq \Sigma_k$ or $\Sigma_h^R \subseteq \Sigma_h$ indicating which cells (or time intervals) should be refined. As already noted in Remark 4.3, the selection of the spatial cells to be refined is done simultaneously on all meshes $\mathcal{T}_h^m$, $m = 0, \ldots, M$.

For the selection of the subsets Σ_k^R or Σ_h^R, several standard approaches are available like error balancing or fixed fraction strategies. However, for the computations in this paper, a quite different approach was used which is described in [32, 33] or [6].

5. Numerical results

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

5.1. Example 1

Let us first consider the following 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 $\boldsymbol{f}$ in such a way that the exact solution $(\boldsymbol{v}, p)^T$ is given by

$$\boldsymbol{v}(t, \boldsymbol{x}) = \begin{pmatrix} \sin(t)\sin^2(\pi x_1)\sin(\pi x_2)\cos(\pi x_2) \\ -\sin(t)\sin(\pi x_1)\cos(\pi x_1)\sin^2(\pi x_2) \end{pmatrix},$$
$$p(t, \boldsymbol{x}) = \sin(t)\sin(\pi x_1)\cos(\pi x_1)\sin(\pi x_2)\cos(\pi x_2).$$

The initial and boundary conditions are set to

$$\begin{aligned} \boldsymbol{v} &= \boldsymbol{0} \quad \text{in } \{\, 0 \,\} \times \Omega, \\ \boldsymbol{v} &= \boldsymbol{0} \quad \text{on } (0,1) \times \partial\Omega. \end{aligned}$$

We aim at computing the functional value

$$J(\boldsymbol{u}) = \frac{1}{2} \int_{\Omega} \left| \boldsymbol{v}(1, \boldsymbol{x}) \right|^2 \, \mathrm{d}\boldsymbol{x}$$

at final time $T = 1$. The exact value can be computed to be

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

First, we present the numerical justification for the splitting of the total discretization error into a temporal and spatial contribution. In Tables 1 and 2, the independence of the temporal error estimator on the refinement of the spatial discretization and vice versa can be seen. 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 subintervals. Also note the very good agreement of the spatial error estimators between both temporal discretizations (columns three and four in Tables 1 and 2) 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.

In Tables 3 and 4, we present the development of the total discretization error $J(\boldsymbol{u}) - J(\boldsymbol{u_{kh}})$ as well as the spatial and temporal error estimators η_h and η_k during an adaptive run with local refinement of the temporal and spatial discretization using dynamic meshes for the cG(1)dG(0) and cG(2)dG(1) discretization, respectively. Here, $N_{\max}$ denotes the number of degrees of freedom of the finest spatial mesh used whereas N_{tot} denotes the

Table 1: 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

N	M	η_h		η_k	
		cG(1)	cG(2)	cG(1)	cG(2)
243	40			$3.8136 \cdot 10^{-04}$	$4.2374 \cdot 10^{-04}$
867	40			$4.1703 \cdot 10^{-04}$	$4.2887 \cdot 10^{-04}$
3267	40	—	—	$4.2620 \cdot 10^{-04}$	$4.2922 \cdot 10^{-04}$
12675	40			$4.2848 \cdot 10^{-04}$	$4.2924 \cdot 10^{-04}$
49923	40			$4.2905 \cdot 10^{-04}$	$4.2924 \cdot 10^{-04}$
3267	10	$1.9876 \cdot 10^{-04}$	$1.5554 \cdot 10^{-06}$		
3267	20	$2.0636 \cdot 10^{-04}$	$1.6150 \cdot 10^{-06}$		
3267	40	$2.1009 \cdot 10^{-04}$	$1.6442 \cdot 10^{-06}$	—	—
3267	80	$2.1194 \cdot 10^{-04}$	$1.6586 \cdot 10^{-06}$		
3267	160	$2.1286 \cdot 10^{-04}$	$1.6656 \cdot 10^{-06}$		

Table 2: 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

N	M	η_h		η_k	
		cG(1)	cG(2)	cG(1)	cG(2)
243	40			$-5.4041 \cdot 10^{-07}$	$-5.7303 \cdot 10^{-07}$
867	40			$-5.6914 \cdot 10^{-07}$	$-5.7818 \cdot 10^{-07}$
3267	40	—	—	$-5.7621 \cdot 10^{-07}$	$-5.7851 \cdot 10^{-07}$
12675	40			$-5.7795 \cdot 10^{-07}$	$-5.7853 \cdot 10^{-07}$
49923	40			$-5.7839 \cdot 10^{-07}$	$-5.7853 \cdot 10^{-07}$
3267	10	$2.1389 \cdot 10^{-04}$	$1.6741 \cdot 10^{-06}$		
3267	20	$2.1379 \cdot 10^{-04}$	$1.6732 \cdot 10^{-06}$		
3267	40	$2.1377 \cdot 10^{-04}$	$1.6731 \cdot 10^{-06}$	—	—
3267	80	$2.1377 \cdot 10^{-04}$	$1.6730 \cdot 10^{-06}$		
3267	160	$2.1377 \cdot 10^{-04}$	$1.6728 \cdot 10^{-06}$		

Table 3: Adaptive refinement on dynamic meshes with equilibration for the cG(1)dG(0) discretization

N_{tot}	$N_{\max}$	M	η_h	η_k	$J(\boldsymbol{u}) - J(\boldsymbol{u_{kh}})$	I_{eff}
2673	243	10	$2.82 \cdot 10^{-03}$	$1.39 \cdot 10^{-03}$	$5.42 \cdot 10^{-03}$	1.29
5655	867	12	$8.26 \cdot 10^{-04}$	$8.33 \cdot 10^{-04}$	$1.96 \cdot 10^{-03}$	1.18
18621	3267	14	$2.25 \cdot 10^{-04}$	$5.04 \cdot 10^{-04}$	$8.05 \cdot 10^{-04}$	1.10
91113	12435	18	$6.32 \cdot 10^{-05}$	$2.71 \cdot 10^{-04}$	$3.57 \cdot 10^{-04}$	1.07
162657	12435	26	$6.07 \cdot 10^{-05}$	$1.41 \cdot 10^{-04}$	$2.08 \cdot 10^{-04}$	1.03
767913	47859	34	$1.94 \cdot 10^{-05}$	$8.51 \cdot 10^{-05}$	$1.03 \cdot 10^{-04}$	0.98
1402389	47859	54	$1.87 \cdot 10^{-05}$	$4.48 \cdot 10^{-05}$	$6.23 \cdot 10^{-05}$	0.98
7419177	177627	82	$6.36 \cdot 10^{-06}$	$2.63 \cdot 10^{-05}$	$3.18 \cdot 10^{-05}$	0.97

total number of degrees of freedom of the space-time discretization. The last column shows the effectivity index I_{eff} which is given by

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

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

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

- "uniform": We apply uniform refinement to the temporal and spatial discretization in each iteration.
- "adaptive": We apply adaptive refinement to the temporal and spatial discretization together with the proposed equilibration strategy. The spatial mesh is fixed on the whole time interval.
- "dynamic": We apply adaptive refinement to the temporal and spatial discretization together with the proposed equilibration strategy. The spatial meshes may vary from subinterval to subinterval.

Table 4: Adaptive refinement on dynamic meshes with equilibration for the cG(2)dG(1) discretization

N_{tot}	$N_{\max}$	M	η_h	η_k	$J(\boldsymbol{u}) - J(\boldsymbol{u_{kh}})$	I_{eff}
5103	243	10	$4.49 \cdot 10^{-04}$	$-1.71 \cdot 10^{-05}$	$3.84 \cdot 10^{-04}$	0.89
6975	867	10	$3.03 \cdot 10^{-05}$	$-2.89 \cdot 10^{-05}$	$4.12 \cdot 10^{-06}$	2.81
27243	3267	12	$1.68 \cdot 10^{-06}$	$-3.47 \cdot 10^{-06}$	$-2.43 \cdot 10^{-06}$	1.35
106167	12675	14	$1.12 \cdot 10^{-07}$	$-6.20 \cdot 10^{-07}$	$-5.12 \cdot 10^{-07}$	1.01
233067	12675	20	$1.21 \cdot 10^{-07}$	$-8.82 \cdot 10^{-08}$	$3.76 \cdot 10^{-08}$	1.15
1028019	49923	24	$1.01 \cdot 10^{-08}$	$-1.77 \cdot 10^{-08}$	$-8.78 \cdot 10^{-09}$	1.15
7651347	197571	40	$7.57 \cdot 10^{-10}$	$-2.56 \cdot 10^{-09}$	$-1.92 \cdot 10^{-09}$	1.07

Even for this example with smooth solution, we achieve a reduction factor of about 50–100 in the degrees of freedom needed for reaching a certain accuracy when using adaptive refinement in time and space compared to uniform meshes. In situations where the dynamics in space are larger than here, one can also achieve a greater 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 depicted in Figure 6. The other discretization leads to similar adaptive refinement at the end of the time interval. This is not very surprising since the functional J only acts at final time $T = 1$ and the solution is mainly driven by the force $\boldsymbol{f}$ and not by problem inherent dynamics.

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

5.2. Example 2

As a second application, we consider the benchmark configuration "Laminar Flow Around a Cylinder", see [34]. The geometry is depicted in Figure 8. Aim of the simulation is the efficient computation of the mean drag-coefficient.

The kinematic viscosity is set to $\nu = 10^{-3}\,\mathrm{m}^2\,\mathrm{s}^{-1}$ while the density is given by $\rho = 1\,\mathrm{kg}\,\mathrm{m}^{-3}$. As initial condition $\boldsymbol{v}(0\,\mathrm{s}, \boldsymbol{x}) = \boldsymbol{0}\,\mathrm{m}\,\mathrm{s}^{-1}$ is chosen.

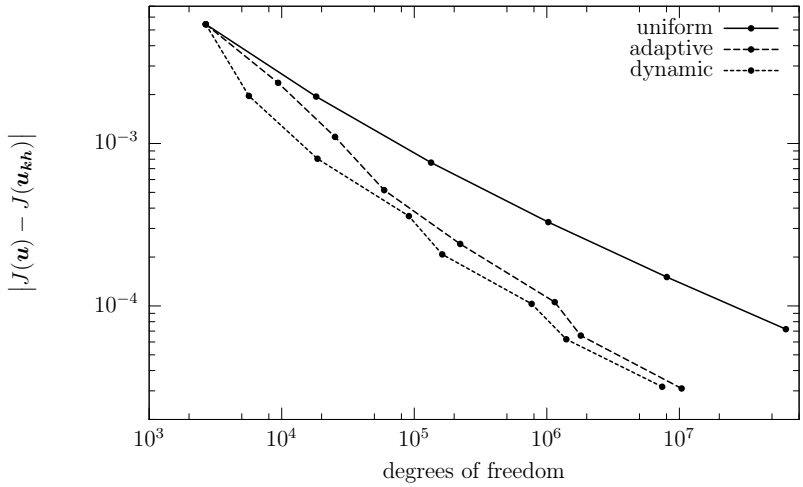


Figure 4: Comparison of the error $\left|J(\boldsymbol{u}) - J(\boldsymbol{u_{kh}})\right|$ for different refinement strategies with the cG(1)dG(0) discretization

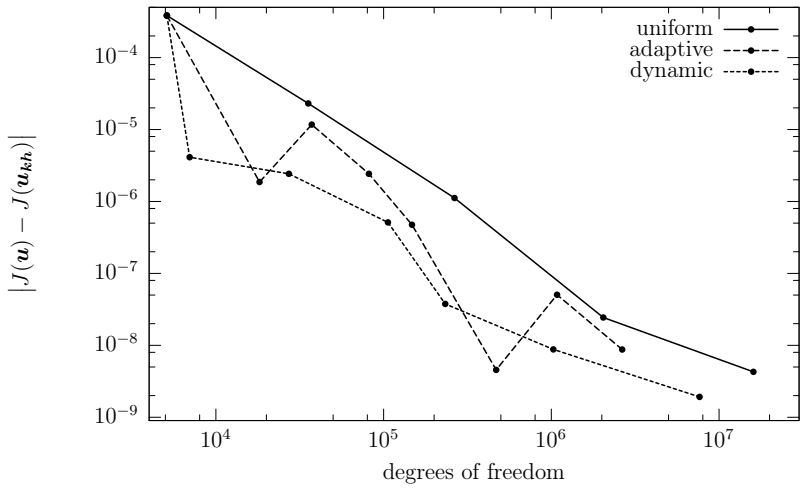


Figure 5: Comparison of the error $\left|J(\boldsymbol{u}) - J(\boldsymbol{u_{kh}})\right|$ for different refinement strategies with the cG(2)dG(1) discretization

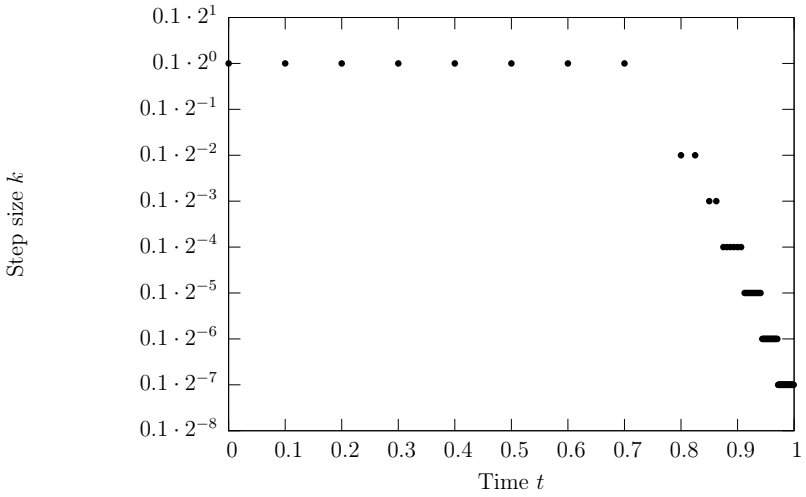


Figure 6: Adaptively determined time step size k with the cG(1)dG(0) discretization

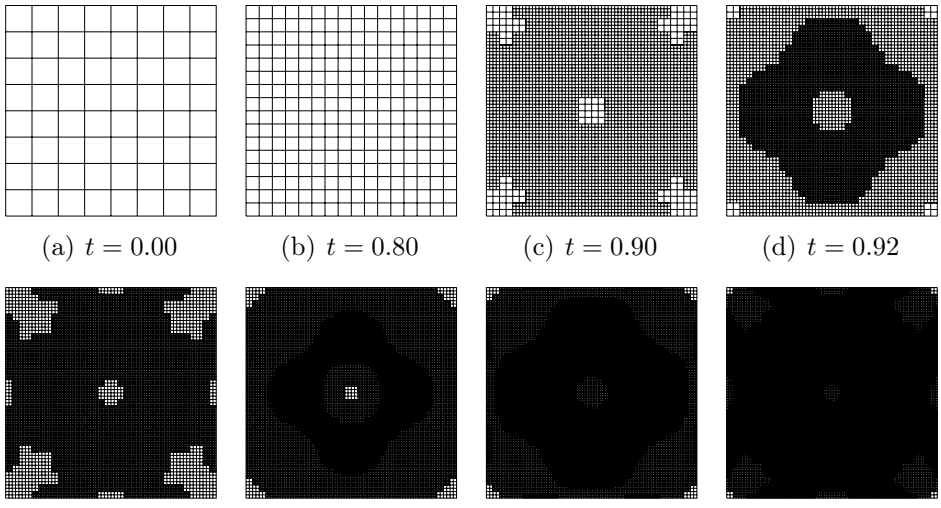


Figure 7: Spatial meshes at different time points obtained with the cG(1)dG(0) discretization

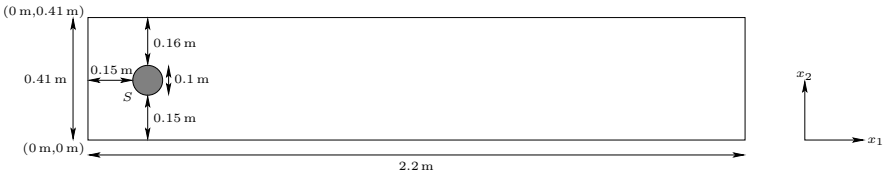


Figure 8: Geometry for the benchmark configuration "Laminar Flow Around a Cylinder"

The inflow condition on the left side of the domain is prescribed as

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

On the outflow boundary on the right side of the computational domain, we apply natural boundary conditions. We refer to [14] for more information on these boundary conditions. On all other boundaries, we prescribe no-slip Dirichlet boundary conditions. The final time is set to $T = 8\,\mathrm{s}$. This setting leads to a Reynolds number $\mathrm{Re}(t) = \frac{\bar{U}(t)D}{\nu}$ based on the mean inflow velocity

$$\bar{U}(t) = \frac{2}{3} v_1(t, 0\,\mathrm{m}, 0.205\,\mathrm{m}) = \sin\Big(\frac{\pi t}{8\,\mathrm{s}}\Big)\,\mathrm{m\,s^{-1}}$$

and the diameter of the obstacle $D = 0.1\,\mathrm{m}$ of $0 \leq \mathrm{Re}(t) \leq 100$ for $0\,\mathrm{s} \leq t \leq 8\,\mathrm{s}$.

Remark 5.1. Due to the nonhomogeneous Dirichlet boundary conditions, the variational formulation has to be modified. Instead of seeking $\boldsymbol{u} = (\boldsymbol{v}, p)^T \in X$ satisfying

$$\int_I \big\{(\partial_t \boldsymbol{v}, \boldsymbol{\psi}) + a(\boldsymbol{u})(\boldsymbol{\varphi})\big\}\,\mathrm{d}t + (\boldsymbol{v}(0) - \boldsymbol{v^0}, \boldsymbol{\psi}(0)) = 0 \quad \forall \boldsymbol{\varphi} = (\boldsymbol{\psi}, \chi)^T \in X, \tag{18}$$

we seek a solution $\boldsymbol{u} = (\boldsymbol{v}, p)^T \in (\boldsymbol{v_\Gamma}, 0)^T + X$ satisfying (18) where $\boldsymbol{v_\Gamma}$ is a divergence-free extension of the Dirichlet boundary conditions.

This modification also influences the derivation of the a posteriori error estimators because now the primal solution no longer is an admissible test function for the dual problem and hence in Lemma 3.1, we have

$$L'(y_1)(e) \neq 0.$$

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

$$\rho^*(\boldsymbol{u}, \boldsymbol{z})(\boldsymbol{u} - \tilde{\boldsymbol{u}}_{\boldsymbol{k}}) \quad \text{and} \quad \rho^*(\boldsymbol{u}_{\boldsymbol{k}}, \boldsymbol{z}_{\boldsymbol{k}})(\boldsymbol{u}_{\boldsymbol{k}} - \tilde{\boldsymbol{u}}_{\boldsymbol{kh}}). \tag{19}$$

However, these terms can be approximated using the same higher order reconstruction techniques as in Section 4.1 to replace (19) by

$$\rho^*(\boldsymbol{I}_{\boldsymbol{k}}\boldsymbol{u}_{\boldsymbol{kh}}, \boldsymbol{I}_{\boldsymbol{k}}\boldsymbol{z}_{\boldsymbol{kh}})(\boldsymbol{\Pi}_{\boldsymbol{k}}\boldsymbol{u}_{\boldsymbol{kh}}) \quad \text{and} \quad \rho^*(\boldsymbol{I}_{\boldsymbol{h}}\boldsymbol{u}_{\boldsymbol{kh}}, \boldsymbol{I}_{\boldsymbol{h}}\boldsymbol{z}_{\boldsymbol{kh}})(\boldsymbol{\Pi}_{\boldsymbol{h}}\boldsymbol{u}_{\boldsymbol{kh}})$$

with some interpolation operators $\boldsymbol{I}_{\boldsymbol{k}}$ and $\boldsymbol{I}_{\boldsymbol{h}}$.

The mean drag-coefficient is given as

$$J_d(\boldsymbol{u}) = -\int_I \left\{ (\partial_t \boldsymbol{v}, \hat{\boldsymbol{\psi}}_{\boldsymbol{d}}) + a(\boldsymbol{u})(\hat{\boldsymbol{\varphi}}_{\boldsymbol{d}}) \right\} \mathrm{d}t \tag{20}$$

with $\hat{\boldsymbol{\varphi}}_{\boldsymbol{d}} = (\hat{\boldsymbol{\psi}}_{\boldsymbol{d}}, 0)^T$ fulfilling

$$\hat{\boldsymbol{\psi}}_{\boldsymbol{d}}\big|_S = \begin{pmatrix} |I|^{-1} C \\ 0 \end{pmatrix}, \quad \hat{\boldsymbol{\psi}}_{\boldsymbol{d}}\big|_{\partial\Omega \setminus S} = \boldsymbol{0}.$$

The constant C therein is chosen as

$$C = \frac{2}{\rho \bar{U}^2 D}.$$

Using the cG(2)dG(1) discretization with uniform refinement of the temporal and spatial discretizations, we obtain for the mean drag-coefficient the values listed in Table 5.

Now 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 $I = (0\,\mathrm{s}, 8\,\mathrm{s})$ are shown in Tables 6 and 7. In these tables, we use the extrapolated value $J_d(\boldsymbol{u}) = 1.6072872$.

We observe how the spatial and the temporal discretization errors are equilibrated and kept balanced under further refinement. Further, we note quite a good agreement of the estimated and the actual discretization error especially on finer discretizations which is indicated by $I_{\text{eff}} \approx 1$.

Table 5: Mean drag-coefficient obtained with the cG(2)dG(1) discretization and uniform refinement

N	M	$J_d(\boldsymbol{u_{kh}})$
2124	80	1.6178974
8088	160	1.5695421
31536	320	1.6048954
124512	640	1.6071242
494784	1280	1.6072465
extrapolated		1.6072872

Table 6: Mean drag-coefficient: Adaptive refinement with equilibration for the cG(1)dG(1) discretization

N	M	η_h	η_k	$J_d(\boldsymbol{u}) - J_d(\boldsymbol{u_{kh}})$	I_{eff}
582	80	$-2.99 \cdot 10^{-01}$	$2.16 \cdot 10^{-04}$	$1.35 \cdot 10^{-01}$	-0.45
1302	80	$-9.15 \cdot 10^{-02}$	$2.80 \cdot 10^{-04}$	$1.54 \cdot 10^{-01}$	-1.69
2280	80	$-7.39 \cdot 10^{-03}$	$3.26 \cdot 10^{-04}$	$7.03 \cdot 10^{-02}$	-9.94
5394	80	$4.29 \cdot 10^{-03}$	$3.96 \cdot 10^{-03}$	$3.64 \cdot 10^{-02}$	4.41
10998	120	$4.78 \cdot 10^{-03}$	$5.82 \cdot 10^{-03}$	$1.11 \cdot 10^{-02}$	1.05
25044	128	$1.64 \cdot 10^{-03}$	$4.76 \cdot 10^{-03}$	$6.89 \cdot 10^{-03}$	1.08
70146	256	$1.14 \cdot 10^{-04}$	$7.48 \cdot 10^{-04}$	$8.34 \cdot 10^{-04}$	0.97
70146	258	$1.14 \cdot 10^{-04}$	$7.32 \cdot 10^{-04}$	$8.15 \cdot 10^{-04}$	0.96
70146	516	$1.22 \cdot 10^{-04}$	$9.85 \cdot 10^{-05}$	$1.68 \cdot 10^{-04}$	0.76
235554	1032	$4.25 \cdot 10^{-05}$	$1.33 \cdot 10^{-05}$	$6.93 \cdot 10^{-05}$	1.24

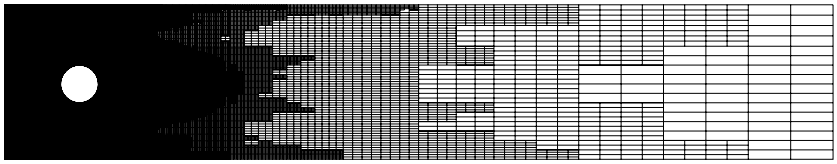
Table 7: Mean drag-coefficient: Adaptive refinement with equilibration for the cG(2)dG(1) discretization

N	M	η_h	η_k	$J_d(\boldsymbol{u}) - J_d(\boldsymbol{u_{kh}})$	I_{eff}
2124	80	$-6.19 \cdot 10^{-02}$	$9.37 \cdot 10^{-04}$	$-1.06 \cdot 10^{-02}$	0.17
5448	80	$2.40 \cdot 10^{-02}$	$8.20 \cdot 10^{-03}$	$5.50 \cdot 10^{-02}$	1.71
11148	160	$2.38 \cdot 10^{-03}$	$4.22 \cdot 10^{-03}$	$5.47 \cdot 10^{-03}$	0.83
27132	252	$-2.29 \cdot 10^{-04}$	$7.79 \cdot 10^{-04}$	$2.09 \cdot 10^{-04}$	0.38
27132	258	$-2.29 \cdot 10^{-04}$	$7.31 \cdot 10^{-04}$	$1.55 \cdot 10^{-04}$	0.31
27132	516	$-1.92 \cdot 10^{-04}$	$9.88 \cdot 10^{-05}$	$-4.06 \cdot 10^{-04}$	4.33
76656	1032	$-3.08 \cdot 10^{-05}$	$1.33 \cdot 10^{-05}$	$-2.52 \cdot 10^{-05}$	1.44

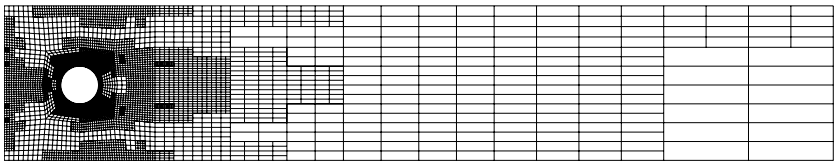
Remark 5.2. The dG(0) discretization in time is not considered in this subsection due to its high numerical dissipation which makes the use of this discretization unfavorable for simulations in computational fluid dynamics.

Remark 5.3. 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, see [35]. However, this would be too costly. The other reason becomes clear if we have a look at Figure 9 where the adaptive spatial meshes corresponding to the last lines in Tables 6 and 7 are presented. We observe that in order to precisely determine the mean drag-coefficient it is not necessary to resolve the whole van 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 further 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 arrive at the conclusion that the use of dynamic spatial meshes does not make sense in this case.

In Figure 10, we show the temporal evolution of the drag-coefficient c_d for all four discretizations considered here. These values correspond to the finest adaptive discretization described in the last lines of Tables 6 and 7 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 note a perfect



(a) cG(1)dG(1) discretization



(b) cG(2)dG(1) discretization

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

Table 8: Maximum drag-coefficient: Comparison with reference values for different discretizations

	$c_{d,\max}^{(\mathrm{ref})} = 2.950921575$		$t^{(\mathrm{ref})}(c_{d,\max}) = 3.93625\,\mathrm{s}$	
Discretization	$c_{d,\max}$	$\left\lvert \frac{c_{d,\max} - c_{d,\max}^{(\mathrm{ref})}}{c_{d,\max}^{(\mathrm{ref})}} \right\rvert$	$t(c_{d,\max})$	$\left\lvert \frac{t(c_{d,\max}) - t^{(\mathrm{ref})}(c_{d,\max})}{t^{(\mathrm{ref})}(c_{d,\max})} \right\rvert$
cG(1)dG(1)	2.950833347	$3.0 \cdot 10^{-5}$	3.9375 s	$3.2 \cdot 10^{-4}$
cG(2)dG(1)	2.950914600	$2.4 \cdot 10^{-6}$	3.9375 s	$3.2 \cdot 10^{-4}$

agreement of all four curves. The corresponding adaptive spatial meshes are the ones already shown in Figure 9. Due to the higher order of the cG(2) discretization compared to the cG(1) discretization in space, the mesh in Figure 9(a) is stronger refined than the one in Figure 9(b).

Even though we aim at efficiently computing the mean drag-coefficient in this example, 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 [36] is given in Table 8.

Figures 11 and 12 show a comparison of different refinement strategies for the cG(1)dG(1) and cG(2)dG(1) discretization, respectively. We use the

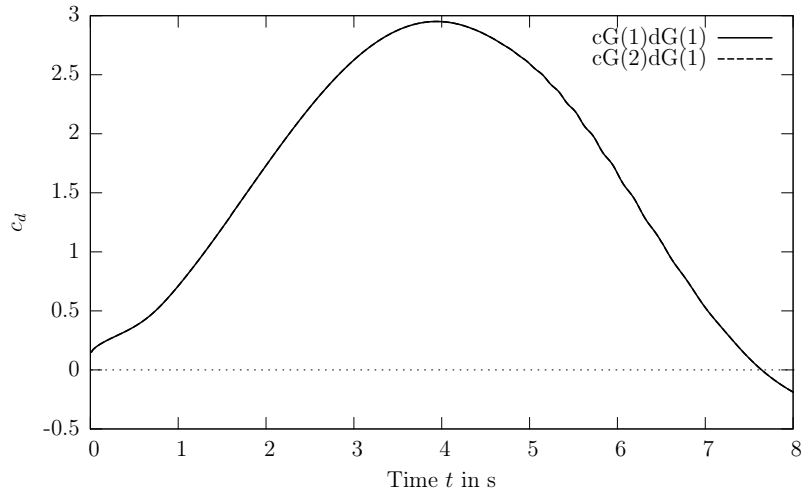


Figure 10: Temporal evolution of the drag-coefficient c_d obtained by different discretizations with adaptive refinement

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 5–15 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 70 for a relative error of 0.1 % in the case of the cG(1)dG(1) discretization.

Conclusions

In this paper we developed a goal-oriented a posteriori error estimator for the nonstationary incompressible Navier-Stokes equations. The error estimator is computable and able to assess the temporal and spatial discretization errors separately. The discretization in time and space was done by Galerkin finite element methods. The proposed method was validated by numerical examples including an established Navier-Stokes benchmark.

References

[1] R. Verfürth, A Review of A Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques, Wiley-Teubner Series Advances in Numerical Mathematics, Wiley-Teubner, New York Stuttgart, 1996.

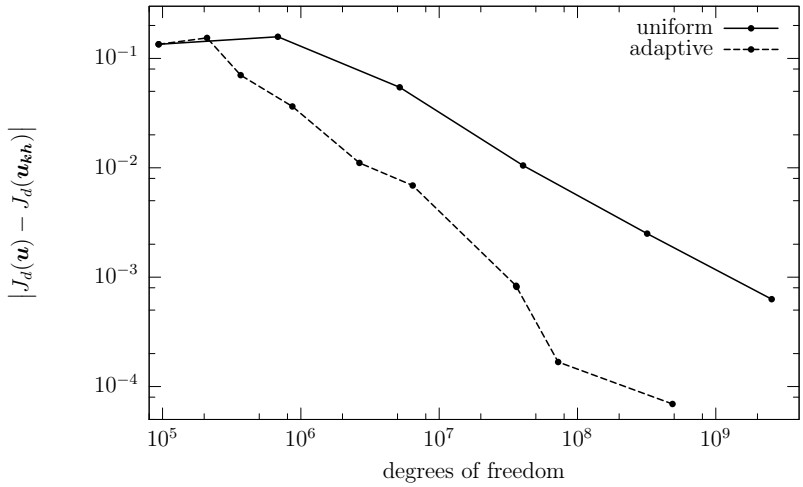


Figure 11: Comparison of the error $\left|J_d(\boldsymbol{u}) - J_d(\boldsymbol{u_{kh}})\right|$ for different refinement strategies with the cG(1)dG(1) discretization

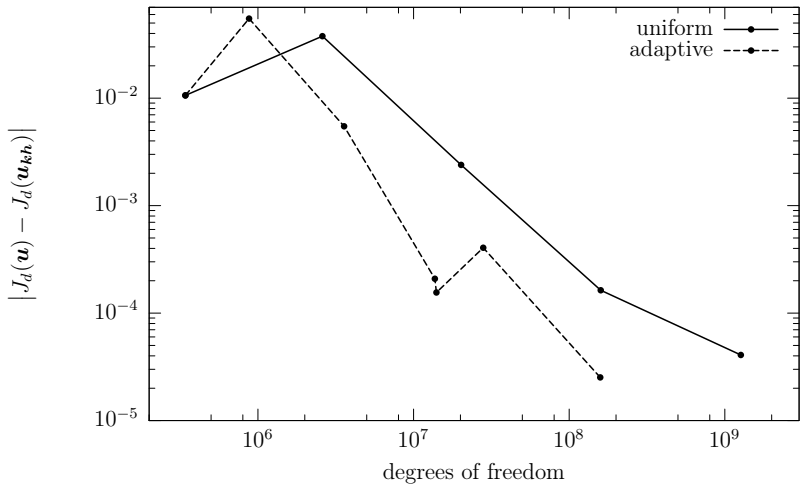


Figure 12: Comparison of the error $\left|J_d(\boldsymbol{u}) - J_d(\boldsymbol{u_{kh}})\right|$ for different refinement strategies with the cG(2)dG(1) discretization

[2] K. Eriksson, D. Estep, P. Hansbo, C. Johnson, Introduction to adaptive methods for differential equations, in: A. Iserles (Ed.), Acta Numerica 1995, volume 4, Cambridge University Press, Cambridge, 1995, pp. 105–158.

[3] D. A. Kay, P. M. Gresho, D. F. Griffiths, D. J. Silvester, Adaptive time-stepping for incompressible flow part II: Navier-Stokes equations, SIAM J. Sci. Comput. 32 (2010) 111–128.

[4] V. John, J. Rang, Adaptive time step control for the incompressible Navier-Stokes equations, Comput. Methods Appl. Mech. Engrg. 199 (2010) 514–524.

[5] E. Bänsch, An adaptive finite-element strategy for the three-dimensional time-dependent Navier-Stokes equations, J. Comput. Appl. Math. 36 (1991) 3–28.

[6] M. Schmich, Adaptive Finite Element Methods for Computing Non-stationary Incompressible Flows, Ph.D. thesis, Universität Heidelberg, Heidelberg, 2009.

[7] M. Schmich, B. Vexler, Adaptivity with dynamic meshes for space-time finite element discretizations of parabolic equations, SIAM J. Sci. Comput. 30 (2008) 369–393.

[8] S. Mittal, A. Ratner, D. Hastreiter, T. E. Tezduyar, Space-time finite element computation of incompressible flows with emphasis on flows involving oscillating cylinders, International Video Journal of Engineering Research 1 (1991) 83–96.

[9] S. Mittal, T. E. Tezduyar, Notes on the stabilized space-time finite-element formulation of unsteady incompressible flows, Comput. Phys. Comm. 73 (1992) 93–112.

[10] M. Behr, T. E. Tezduyar, Finite element solution strategies for large-scale flow simulations, Comput. Methods Appl. Mech. Engrg. 112 (1994) 3–24.

[11] D. N'dri, A. Garon, A. Fortin, Incompressible Navier-Stokes computations with stable and stabilized space-time formulations: A comparative study, Comm. Numer. Methods Engrg. 18 (2002) 495–512.

[12] J. Hoffman, Efficient computation of mean drag for the subcritical flow past a circular cylinder using General Galerkin G2, Internat. J. Numer. Methods Fluids (2008). To appear.

[13] R. Temam, Navier-Stokes Equations: Theory and Numerical Analysis, AMS Chelsea Publishing, Providence, Rhode Island, 2001.

[14] J. G. Heywood, R. Rannacher, S. Turek, Artificial boundaries and flux and pressure conditions for the incompressible Navier-Stokes equations, Internat. J. Numer. Methods Fluids 22 (1996) 325–352.

[15] P. G. Ciarlet, The Finite Element Method for Elliptic Problems, North-Holland Publishing Company, Amsterdam New York Oxford, first edition, 1987.

[16] G. F. Carey, J. T. Oden, Finite Elements. Volume III. Computational Aspects, The Texas Finite Element Series, Prentice-Hall, Inc., Englewood Cliffs, 1984.

[17] K. Eriksson, D. Estep, P. Hansbo, C. Johnson, Computational Differential Equations, Cambridge University Press, 1996.

[18] P. Hood, C. Taylor, Navier-Stokes equations using mixed interpolation, in: J. T. Oden, O. C. Zienkiewicz, R. H. Gallagher, C. Taylor (Eds.), Finite Element Methods in Flow Problems, University of Alabama, Huntsville, 1974, pp. 121–132.

[19] R. Becker, M. Braack, A modification of the least-squares stabilization for the Stokes equations, Calcolo 38 (2001) 173–199.

[20] R. Becker, M. Braack, A two-level stabilization scheme for the Navier-Stokes equations, in: M. Feistauer, V. Dolejší, P. Knobloch, K. Najzar (Eds.), Numerical Mathematics and Advanced Applications. ENUMATH 2003, Springer-Verlag, Heidelberg, 2004, pp. 123–130.

[21] L. P. Franca, S. L. Frey, Stabilized finite element methods: II. The incompressible Navier-Stokes equations, Comput. Methods Appl. Mech. Engrg. 99 (1992) 209–233.

[22] M. Braack, E. Burman, V. John, G. Lube, Stabilized finite element methods for the generalized Oseen equations, Comput. Methods Appl. Mech. Engrg. 196 (2007) 853–866.

[23] R. Becker, R. Rannacher, An optimal control approach to a posteriori error estimation in finite element methods, in: A. Iserles (Ed.), Acta Numerica 2001, volume 10, Cambridge University Press, Cambridge, 2001, pp. 1–102.

[24] J. Wloka, Partielle Differentialgleichungen, B. G. Teubner, Stuttgart, 1982.

[25] A. Griewank, Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation, Optim. Methods Softw. 1 (1992) 35–54.

[26] M. Berggren, R. Glowinski, J. L. Lions, A computational approach to controllability issues for flow-related models. (I): Pointwise control of the viscous Burgers equation, Int. J. Comput. Fluid Dyn. 7 (1996) 237–252.

[27] A. Walther, A. Griewank, Advantages of binomial checkpointing for memory-reduced adjoint calculations, in: M. Feistauer, V. Dolejší, P. Knobloch, K. Najzar (Eds.), Numerical Mathematics and Advanced Applications. ENUMATH 2003, Springer-Verlag, Heidelberg, 2004, pp. 834–843.

[28] D. Meidner, Adaptive Space-Time Finite Element Methods for Optimization Problems Governed by Nonlinear Parabolic Systems, Ph.D. thesis, Universität Heidelberg, Heidelberg, 2007.

[29] C. Carstensen, R. Verfürth, Edge residuals dominate a posteriori error estimates for low order finite element methods, SIAM J. Numer. Anal. 36 (1999) 1571–1587.

[30] R. Becker, R. Rannacher, A feed-back approach to error control in finite element methods: Basic analysis and examples, East-West J. Numer. Math. 4 (1996) 237–264.

[31] M. Braack, A. Ern, A posteriori control of modeling errors and discretization errors, Multiscale Model. Simul. 1 (2003) 221–238.

[32] T. Richter, Funktionalorientierte Gitteroptimierung bei der Finite-Elemente-Approximation elliptischer Differentialgleichungen, Diploma thesis, Universität Heidelberg, Heidelberg, 2001.

[33] T. Richter, Parallel Multigrid Method for Adaptive Finite Elements with Application to 3D Flow Problems, Ph.D. thesis, Universität Heidelberg, Heidelberg, 2005.

[34] M. Schäfer, S. Turek, Benchmark computations of laminar flow around a cylinder, in: E. H. Hirschel (Ed.), Flow Simulation with High-Performance Computer II, volume 52 of *Notes on Numerical Fluid Mechanics*, Vieweg, Braunschweig Wiesbaden, 1996, pp. 547–566.

[35] M. Besier, W. Wollner, On the Dependence of the Pressure on the Time Step in Incompressible Flow Simulations on Varying Spatial Meshes, Preprint, Universität Heidelberg, 2010.

[36] V. John, Reference values for drag and lift of a two-dimensional time-dependent flow around a cylinder, Internat. J. Numer. Methods Fluids 44 (2004) 777–788.