Published in **Computers and Fluids** Vol. 35, pp. 372-392, 2006

Solutions of 3D Navier-Stokes benchmark problems with adaptive finite elements

M. Braack T. Richter

This paper presents a numerical study of 3D Navier-Stokes benchmark problems defined within the DFG high-priority research program in 1996. Specifically, we investigate the accuracy of an equal-order finite element method based on piecewise quadratic shape functions with local projections stabilization on locally refined meshes for stationary laminar flows around an obstacle with circular and square cross-section. It turns out that on globally refined meshes the new stabilization method is comparable to Q_2/P_1^{disc} element which was the best one in recent investigations of John [1]. Furthermore, on locally refined meshes we are able to produce reference values for the geometry with singularities (square cross section) which were still unknown up to now.

1 Introduction

Under the DFG Priority Research Program 'Flow Simulation on High Performance Computers' a set of benchmark problems has been defined by Schäfer & Turek [2]. The task was to calculate different coefficients of a three dimensional flow around an obstacle. Two different obstacles - a cylinder with circular and with square cross-section - are considered. In both cases the inflow velocity is specified and yields a Reynolds number $Re = 20$ leading to a steady flow. The requested quantities are the drag coefficient, the lift coefficient and the pressure difference in two points on the obstacle. Although the problem has been formulated a couple of years ago accurate reference solutions are only determined for the geometry with circular cross-section by John [1]. A reference solution for the square cross-section is still missing, which is probably connected to the fact that the corners on the square destroy the regularity of the solution.

In this work, we present numerical results for these two benchmark configurations obtained with a non-standard stabilized finite element scheme, namely with local projection stabilization (LPS), see Becker & Braack [3, 4], Braack & Burman [5] and [6]. In contrast to previous work, the discretization uses piecewise triquadratic elements and local projection of parts of the residual on bilinears. The determination of a reference solution for the square cross-section was possible by local mesh refinement controlled by a posteriori error estimation. Here, we applied the dual-weighted residual method of Becker & Rannacher [7]. It turns out, that we confirm the results in [1] for the configuration with the smooth obstacle (circular cylinder); for instance, up to 6 leading digits for the drag. Moreover, for the singular geometry (square cylinder), we present improved reference values for drag, lift and pressure drop; e.g. for the drag up to 4 digits correct.

The outline of this paper is as follows. We begin by recalling the benchmark configurations and the determination of the interesting quantities. In Section 3, the finite element discretization is developed. The design of the adaptive algorithm and the a posteriori error estimation is presented in Section 4. The numerical results are given in Section 5. We finish with a short summary.

2 The benchmark problems for 3D laminar flows

We consider flows around obstacles which were defined in [2] as benchmark problems within the DFG high-priority research program "Flow simulation with high-performance computers". The governing equations for the velocity v and pressure p are the Navier-Stokes equations for steady incompressible flow in the domain $\Omega \subset \mathbb{R}^3$:

$$\begin{aligned} -\mu\Delta v + (v\cdot\nabla)v + \nabla p &= 0\,, \\ \operatorname{div} v &= 0\,, \end{aligned}$$

equipped with boundary conditions on $\partial\Omega = \Gamma_{\text{in}} \cup \Gamma_{\text{wall}} \cup \Gamma_{\text{out}}$:

$$\begin{aligned} v &= \widehat{v} \quad \text{on } \Gamma_{\text{in}}\,, \\ v &= 0 \quad \text{on } \Gamma_{\text{wall}}\,, \\ \mu\frac{\partial v}{\partial n} - pn &= 0 \quad \text{on } \Gamma_{\text{out}}\,, \end{aligned} \tag{1}$$

where Γ_{in} is the inflow boundary, Γ_{wall} are solid walls, and Γ_{out} is the outflow part of the boundary. The outflow condition (1) was not mandatory in the definition of the benchmark problem but could be chosen by the participants. The domain Ω is a channel with an obstacle. The height of the channel is $H = 0.41\,m$, the length is $L = 2.5\,m$. In the first configuration, the obstacle is a cylinder with

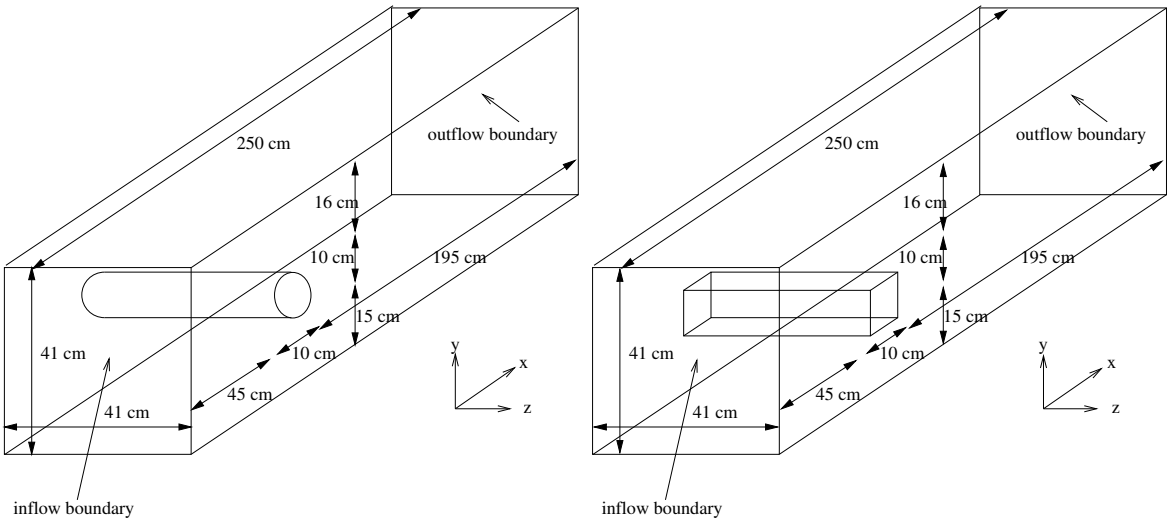

Figure 1: Configurations of the benchmark problems. The obstacle is a cylinder with circular cross-section (configuration 1, left) and square cross-section (configuration 2, right).

diameter $D = 0.1\,m$. In the second one, the obstacle consists of a square cross-section with the same diameter D. The exact (non-symmetric) positions of these obstacles are displayed in Figure 1.

For both configurations, the kinematic viscosity is given by $\mu = 10^{-3} m^2/s$. The inflow velocity field $\widehat{v} = \{\widehat{v}_1, 0, 0\}$ is prescribed by the component in $x-$direction $\widehat{v}_1 = v^+ w(y) w(z)$ with the parabolic function $w(y) = 4y(H-y)/H^2$ and the peak velocity $v^+ = 0.45\,m/s$. Both configurations have the Reynolds number $Re = 20$ based on ν, D and the mean inflow velocity $\overline{v} = 0.2\,m/s$.

The quantities to compute are the following three:

- The pressure difference Δp between the points $p_1 = \{0.55, 0.2, 0.205\}$ and $p_2 = \{0.45, 0.2, 0.205\}$:

$$\Delta p = p_2 - p_1\,.$$

- The drag coefficient c_{drag} of the obstacle is given by an integral over the surface $S \subset \Gamma_{\text{wall}}$ of the obstacle:

$$c_{\text{drag}} = C \int_S \left(\nu \frac{\partial v_t}{\partial n} n_y - p n_x \right) ds\,, \quad C = \frac{2}{DH\overline{v}^2} = \frac{500}{0.41}\,.$$

  Here, the vector $n = \{n_x, n_y, n_z\}$ denotes the unit normal vector on the surface S pointing into Ω, and v_t denotes the tangential velocity of v on tangent $t = \{n_y, -n_x, 0\}$.

- The lift coefficient of the obstacle is defined very similar:

$$c_{\text{lift}} = -C \int_S \left(\nu \frac{\partial v_t}{\partial n} n_x + p n_y \right) ds\,.$$

In summary, the requested quantities are the three real numbers $\Delta p, c_{\text{drag}}, c_{\text{lift}}$ for each of the two configurations. These can be considered as a vector-valued functional $j(u)$ of the solution $u = \{v, p\}$:

$$j(u) := (\Delta p, c_{\text{drag}}, c_{\text{lift}})\,.$$

For the obstacle with the circular cross-section (configuration 1), reference solutions are published in [1]. In that work, several discretizations are compared with respect to accuracy. Following the

#cells	dof	Δp	c_{drag}	c_{lift}
245 760	7 035 840	0.170403	6.185234	9.40479e-3
1 966 080	55 666 560	0.170779	6.185327	9.40122e-3
extrapolated			6.185329	9.40098e-3

Table 1: Reference values published in [1] for the circular cross-section.

#cells	dof	Δp	c_{drag}	c_{lift}
480	18 720	0.188771	6.250365	
3 840	136 000	0.178702	6.172750	1.02332e-2
30 720	983 040	0.173744	6.184551	9.46862e-3
245 760	7 864 320	0.171999	6.185323	9.43526e-3
1 966 080	62 914 560	0.171342	6.185331	9.40136e-3

Table 2: Values obtained in this work for the circular cross-section on structured meshes.

conclusion of the author, the most accurate finite element discretization for this problem under their consideration consists of piecewise triquadratic elements for the velocities (Q_2) and discontinuous piecewise linear pressure (P_1^{disc}). Their computation has been performed on structured meshes by the use of a parallel computer. The obtained values on the finest meshes for Q_2/P_1^{disc} and extrapolated values are recapitulated in Table 1. Their finest mesh contains about 56 million degrees of freedom (dof). The underlined digits are those ones which can be considered as reliable based on the convergence history. For the pressure drop, only the first two leading digits are ensured: $\Delta p = 0.17$. The drag coefficient was the "easiest" quantity to be computed: the first four digits are reliable: $c_{\text{drag}} = 6.185$. The conjecture due to extrapolation based on the finest meshes is a value of $c_{\text{drag}} = 6.1853$. If this is true, on the finest mesh also the fifth digit is correct. For the lift coefficient c_{lift}, the first three leading digits are known, $c_{\text{lift}} = 9.40 \cdot 10^{-3}$.

For the square cross-section (configuration 2), the accuracy of simulations on structured meshes is only reliable up to the first two digits. For instance, the drag coefficient is believed to be $c_{\text{drag}} = 7.7$ for configuration 2.

In this work, we will apply local mesh refinement with stabilized triquadratic elements (Q_2) for both, velocity and pressure. The stabilization takes advantage of the stable subspace Q_2/Q_1. In order to validate that this discretization is at least as accurate as the established one Q_2/P_1^{disc} in [1], we list our values for structured meshes in Table 2. The meshes used are exactly the same as those which were used to get the values in Table 1. The number of degrees of freedom are moderately higher for the equal-order discretization compared to the one with discontinuous pressure. However, the accuracy is pretty much the same. For the pressure drop Δp, the first two digits are reliable. With respect to the drag coefficient, also the fifth digit can now be considered as reliable $c_{\text{drag}} = 6.1853$, which confirms the extrapolation in [1].

This is a good starting point to analyze the effect of local mesh refinement in order to obtain similar values on meshes with less mesh points. Beyond this, we explore the far more difficult test case of the square cross-section (configuration 2). The solution can be found in a Sobolev space $H^{1+\alpha}(\Omega)$ with $0 < \alpha \ll 1$. Therefore, even for quadratic elements, the convergence order (on globally refined meshes) can be expected only to be about $O(h)$. For completeness, in Table 3 we also gather the values obtained with stabilized Q_2/Q_2 elements on structured meshes. In the remainder of this work, we will enhance the accuracy by the use of a posteriori error estimation.

#cells	dof	Δp	c_{drag}	c_{lift}
78	3 696	0.183495	13.31491	-5.88042e-2
624	24 544	0.208050	8.044496	1.04015e-1
4 992	177 600	0.177370	7.975926	7.82310e-2
39 936	1 348 480	0.176165	7.787849	6.78254e-2
319 488	10 787 840	0.175759	7.757928	6.86428e-2
2 555 904	86 302 720	0.175677	7.761191	6.88130e-2

Table 3: Values obtained in this work for the square cross-section on structured meshes.

3 Finite element discretization

We prolongate the boundary conditions $\widehat{v}$ to the no-slip boundaries by setting $\widehat{v} = 0$ on Γ_{wall}. The resulting Dirichlet condition is

$$v = \widehat{v} \quad \text{on } \Gamma_{\text{dir}} := \Gamma_{\text{in}} \cup \Gamma_{\text{wall}} .$$

Furthermore, we set $V := H_0^1(\Omega)^3$ and the space for the velocity we set to $\widehat{V} := \widehat{v} + V$, i.e., the affine subspace of the Sobolev space $H^1(\Omega)^3$ with traces according to the Dirichlet boundaries $\widehat{v}$. The pressure p is supposed to be in the space $Q := L^2(\Omega)$. Velocities and pressure are assembled together in the pair $u = (v, p) \in \widehat{X} := \widehat{V} \times Q$.

The partial differential equation to be solved can be written in short operator form:

$$Lu = f ,$$

with a nonlinear operator on Hilbert spaces $L : \widehat{X} \to H^{-1}(\Omega)^3 \times L^2(\Omega)$, together with the Neumann-type outflow boundary condition for u on Γ_{out}. The right-hand side f vanishes for this benchmark problem.

3.1 Variational formulation.

The space for the test functions $\varphi = (\psi, \xi)$ is $X = V \times Q$. We define the following semilinear form acting on $\widehat{X} \times X$ for the variable $u = (v, p)$:

$$a(u)(\varphi) \quad := \quad (\mu \nabla v, \nabla \psi) + ((v \cdot \nabla)v, \psi) - (p, \operatorname{div} \psi) + (\operatorname{div} v, \xi) ,$$

where the parentheses stand for the usual scalar products in Q and V, respectively. The semilinear form $a(\cdot)(\cdot)$ is nonlinear in the first argument and linear in the second one. The benchmark problem under consideration in variational form now reads

$$u \in \widehat{X} : \quad a(u)(\varphi) = 0 \quad \forall \varphi \in X .$$

In this formulation, the outflow condition (1) is built in due to integration by parts of the viscous term and the pressure gradient.

For the discretization we use a conforming equal-order Galerkin finite element method defined on hexaedral meshes $\mathcal{T}_h = \{K\}$ over the computational domain $\Omega \subset \mathbb{R}^3$, with cells denoted by K. The mesh parameter h is defined as a cell-wise constant function by setting $h_{|K} = h_K$ and h_K is the diameter of K. The straight parts which make up the boundary ∂K of a cell K are called *faces*.

A mesh $\mathcal{T}_h$ is called regular, if it fulfills the standard conditions for shape-regular finite element meshes, see e.g. Ciarlet [8]. However, in order to ease the mesh refinement we allow the cells to have

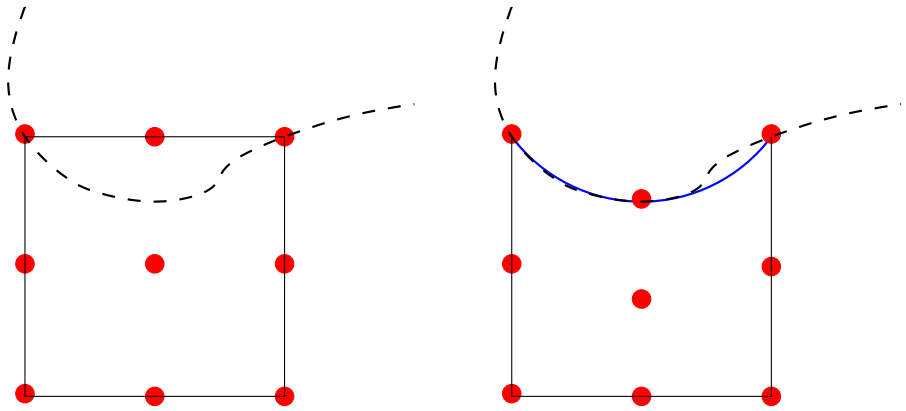

Figure 2: Two possibilities for Q_2 elements on curved boundaries. Without (left) and with (right, used in this work) adjustment of the internal degrees of freedom.

nodes, which lie on midpoints of faces of neighboring cells. But at most one such hanging node is permitted for each face.

For computational domains Ω with piecewise quadratic boundaries, the discrete function space $Q_h \subset Q$ consists of continuous, piecewise triquadratic functions (Q_2-elements) for all unknowns,

$$Q_h = \{\varphi_h \in C(\overline{\Omega});\, \varphi_h|_K \in Q_2\}\,,$$

where Q_2 is the space of functions obtained by transformations of (isoparametric) triquadratic polynomials on a fixed reference unit cell $\hat{K}$. For a detailed description of this standard construction, see [8] or Johnson [9].

The case of hanging nodes requires some additional remarks. There are no degrees of freedom corresponding to these irregular nodes and the value of the finite element function is determined by pointwise interpolation. This implies continuity and therefore global conformity, i.e., $Q_h \subset Q$. The discrete space for the velocities is $V_h := (Q_h)^3 \subset V$, and $\widehat{V}_h := \widehat{v} + V_h$. The discrete product spaces are $X_h := V_h \times Q_h$, and $\widehat{X}_h := \widehat{V}_h \times Q_h$.

Since we apply triquadratic finite elements to a geometry with a curved boundary, we have to be careful with the boundary approximation. In Figure 2, we display two possible distributions of the degrees of freedom. The possibility shown in the left-hand side in the figure may lead to the wrong order and deteriorate the accuracy of the finite element ansatz. We adjust the inner degrees of freedom in the Q_2 ansatz to the boundary (right-hand side of Figure 2). The boundary is approximated by a quadratic polynomial. However, for curved boundaries the method is not necessarily conformal, i.e., $Q_h \not\subset Q$.

The Galerkin formulation of the Navier-Stokes equation with equal-order finite elements is not stable since the discrete inf-sup condition is violated. Furthermore, at higher Reynolds numbers, the advective term becomes unstable for pure Galerkin methods. In order to overcome this limitation, stabilization techniques can be used. We use local projection stabilization (LPS) as introduced in [3, 4] for both types for stabilization. This will be presented in the following subsection.

3.2 Local projection stabilization for Q_2 elements.

The semilinear form $a_h(u_h)(\phi)$ is given by a sum of the Galerkin part and stabilization terms:

$$a_h(u_h)(\varphi) \;:=\; a(u_h)(\varphi) + s_h(u_h)(\varphi)\,.$$

The resulting discrete equation now reads

$$u_h \in \widehat{X}_h : \quad a_h(u_h)(\varphi) = 0 \quad \forall \varphi \in X_h \,.$$

The term $s_h(u_h)(\varphi)$ was initially formulated for piecewise bilinear elements. We modify this stabilization technique in the following form: By $Q'_h \subset Q_h$ we denote the finite element space of piecewise trilinear elements (Q_1) on the same mesh $\mathcal{T}_h$. Furthermore, we denote by $I_h : Q_h \to Q'_h$ the nodal interpolation onto piecewise trilinears, and by $\pi_h : Q_h \to Q_h$ the projection

$$\pi_h \xi := \xi - I_h \xi \,,$$

which yields the quadratic fluctuation with respect to piecewise trilinears. In order to avoid too many notations, we use the notation π_h also for the projection operator of the vector-valued velocities, $\pi_h : V_h \to V_h$ and $\pi_h : \widehat{V}_h \to \widehat{V}_h$.

As already mentioned, the term $s_h(u_h)(\varphi)$ has to handle two types of instabilities. Therefore, for a test function $\varphi = (\psi, \xi) \in X_h$ we use a sum of two terms:

$$s_h(u_h)(\varphi) := s_h^p(p_h)(\xi) + s_h^v(v_h)(\psi) \,,$$

where the first one stabilizes the stiff velocity-pressure coupling, whereas the second one becomes important in convection-dominated regimes.

The proposed stabilization is consistent in the sense that the introduced terms vanish for $h \to 0$. Numerical experience has shown that for smooth solutions, the introduced perturbation is even of higher-order than the discretization error which is supposed to be a consequence of superconvergence. For an interpretation of LPS in the context of a variational multiscale approach we refer to [5].

3.2.1 Pressure-velocity stabilization

For equal-order finite elements, the Galerkin formulation of the Stokes system for the pressure p and velocity v is known to be unstable, since the stiff pressure-velocity coupling for incompressible flows enforces spurious pressure modes due to the saddle-point structure of the Stokes system. The added stabilization term which damps spurious pressure modes is of the form

$$s_h^p(p_h)(\xi) \quad := \quad \sum_{K \in \mathcal{T}_h} \alpha_K \int_K (\nabla \pi_h p_h)(\nabla \pi_h \xi) \, dx \,, \tag{2}$$

with weights $\alpha_K = \alpha_0 h_K^2 / \mu$ depending on the mesh size h_K of cell K and the viscosity μ. The definition of h_K on unstructered meshes is given by the determinant of the transformation $T_K : \widehat{K} \to K$ from the reference unit cell $\widehat{K}$ to K: $h_K := |\det(T_K(\xi_m))|^{1/3}$, where ξ_m denotes the center of $\widehat{K}$. The parameter α_0 is usually chosen between 0.05 and 0.2. The stabilization term (2) acts as a diffusion term on the fine-grid scales of the pressure. The scaling proportional to h_K^2 gives stability of the discrete equations and maintains accuracy. This type of stabilization is introduced in [3] for the Stokes equation. Therein, a stability proof and an error analysis is given. The same stabilization is now applied to the Navier-Stokes equations, see [4]. Note, that π_h vanishes for trilinear test functions on $\mathcal{T}_h$, and therefore, the stabilization vanishes for $\xi \in Q'_h$.

3.2.2 Convection stabilization

The stabilization term added to the Galerkin formulation reads

$$s_h^v(v_h)(\psi) \quad := \quad \sum_{K \in \mathcal{T}_h} \delta_K \int_K ((v_h \cdot \nabla)\pi_h v_h)((v_h \cdot \nabla)\pi_h \phi) \, dx \,, \tag{3}$$

where the cell-wise coefficients δ_K depend on the local balance of convection and diffusion:

$$\delta_K := \frac{\delta_0 h_K^2}{6\mu + h_K \|v_h\|_K} .$$

Here, the quantity $\|v_h\|_K$ is the cell-wise value for the convection. The parameter δ_0 is a fixed constant, usually chosen in the range $0.2 \leq \delta_0 \leq 0.4$. This type of convection stabilization can be considered as a specification of the work of Guermond [10] to triquadratic elements with a special filter.

For high Reynolds number, the optimal choice for α_0 is the same as for δ_0, see [5]. However, the benchmark configurations considered here, are at low Reynolds number.

3.2.3 Dependency on stabilization parameters

In order to document the dependency of accuracy with respect to the stabilization parameters α_0 and δ_0, we perform a couple of runs of configuration I (circular-cross section) with $\alpha_0, \delta_0 \in \{0.05, 0.1, 0.2, 0.5, 1\}$.

In Table 4, the error in c_{drag} for a mesh with 30 720 cells is listed. Obviously, the dependency on α_0 is very low. This can be seen by analyzing one column in the table: Although α_0 varies by a factor of 20, the error in drag varies my a factor of about 2 at most. The difference of the error for $\delta_0 = 0.05$ and for $\delta = 1.$ is also less than 30 %. However, since the sign of the error changes for $\delta_0 = 0.05$ and $\delta = 1$, one may find a configuration of parameters for which the error even vanishes.

α/δ	0.05	0.1	0.2	0.5	1.
0.05	$1.72e^{-3}$	$1.36e^{-3}$	$7.82e^{-4}$	$-3.41e^{-4}$	$-1.45e^{-3}$
0.10	$1.57e^{-3}$	$1.21e^{-3}$	$6.29e^{-4}$	$-4.85e^{-4}$	$-1.58e^{-3}$
0.20	$1.48e^{-3}$	$1.11e^{-3}$	$5.25e^{-4}$	$-5.85e^{-4}$	$-1.67e^{-3}$
0.50	$1.40e^{-3}$	$1.03e^{-3}$	$4.45e^{-4}$	$-6.65e^{-4}$	$-1.76e^{-3}$
1.00	$1.37e^{-3}$	$9.98e^{-4}$	$4.13e^{-4}$	$-6.98e^{-4}$	$-1.78e^{-3}$

Table 4: Error in c_{drag} for configuration 1 on a mesh with 30 720 cells for different choices of stabilization parameters α_0 and δ_0. For instance, one row corresponds to α_0 fixed and δ_0 variable.

3.2.4 Evaluation of surface integrals

The finite element method allows for evaluation of surface integrals of type c_{drag} and c_{lift} by an integral over the whole domain Ω. The advantage is twofold: At first, the order of accuracy is enhanced substantially (4-th order for the used triquadratic elements in the case of the circular cylinder). At second, the implementation is simplified since routines for computing the nonlinear form $a(u_h)(\varphi)$ are used. This evaluation technique was also used in [1]. However, since this evaluation technique is not widely known, we shortly present this technique.

The drag coefficient can also be expressed by (the constant C in the definition of the drag is set here for simplicity to 1)

$$c_{\text{drag}} \quad = \quad \int_S \left(\nu \frac{\partial v_x}{\partial n} - p n_x \right) , \tag{4}$$

because it holds ($t = \{n_y, -n_x, 0\}$):

$$\frac{\partial v_t}{\partial n} n_y \quad = \quad \frac{\partial v_x}{\partial n} n_y^2 - \frac{\partial v_y}{\partial n} n_x n_y$$

$$
\begin{aligned}
&= \frac{\partial v_x}{\partial n} - n_x \left(\frac{\partial v_x}{\partial n} n_x - \frac{\partial v_y}{\partial n} n_y \right) \\
&= \frac{\partial v_x}{\partial n} - n_x \operatorname{div} v \\
&= \frac{\partial v_x}{\partial n} .
\end{aligned}
$$

In order to explain the principal idea of the evaluation of (4) we take the definition of $a(u)(\hat{\varphi})$ with a function $\hat{\varphi}$ which is zero in all components except for the velocity component in $x-$direction denoted by $\hat{\varphi}_1$. Even if this component does not fulfill the boundary condition, i.e., $\hat{\varphi} \notin X$, we obtain by integration by parts:

$$
\begin{aligned}
a(u)(\hat{\varphi}) &= (\nu \nabla v_x, \nabla \hat{\varphi}_1) + ((v \cdot \nabla) v_x, \hat{\varphi}_1) - (p, \partial_x \hat{\varphi}_1) \\
&= (-\nu \Delta v_x + (v \cdot \nabla) v_x + \partial_x p, \hat{\varphi}_1) + \int_{\partial\Omega} \left(\nu \frac{\partial v_x}{\partial n} - p n_x \right) \hat{\varphi}_1 .
\end{aligned}
$$

If we take for u the strong solution of the Navier-Stokes equations (especially $-\nu \Delta v_x + (v \cdot \nabla) v_x + \partial_x p = 0$) and as test function for the horizontal velocity component $\hat{\varphi}_1|_{\partial\Omega \setminus S} = 0$ and $\hat{\varphi}_1|_S = 1$ we obtain the drag coefficient as given by (4):

$$
a(u)(\hat{\varphi}) = c_{\mathrm{drag}} . \tag{5}
$$

This property can be used numerically by taking $\hat{\varphi}_1$ as a Q_2 finite element function with the Dirichlet values on $\partial\Omega$ as mentioned above. The discrete drag $c_{drag,h}$ is evaluated by

$$
c_{drag,h} := a(u_h)(\hat{\varphi}) .
$$

For ease of presentation, we omit here the stabilization term $s_h(\cdot)(\cdot)$. The extension to stabilized methods is straight forward, see [6]. For configuration 1 of the benchmark problems under discussion it turns out that the error in the drag becomes

$$
c_{\mathrm{drag}} - c_{drag,h} = O(h^4) .
$$

The argument to obtain this uses the Gateaux derivative $a'(\xi)(u - u_h, \cdot)$ of $a(\cdot, \cdot)$ at a certain point $\xi := \lambda u + (1 - \lambda) u_h$, $0 \le \lambda \le 1$, and in direction of $u - u_h$:

$$
c_{\mathrm{drag}} - c_{drag,h} = a(u)(\hat{\varphi}) - a(u_h)(\hat{\varphi}) = a'(\xi)(u - u_h, \hat{\varphi}) .
$$

Since $a(u)(\varphi) = a(u_h)(\varphi) = 0$ for all $\varphi \in X_h$ it holds also:

$$
c_{\mathrm{drag}} - c_{drag,h} = a'(\xi)(u - u_h, \hat{\varphi} + \varphi) \quad \forall \varphi \in X_h .
$$

Now, we introduce the continuous dual solution $z \in \widehat{X}$ given by the following continuous problem:

$$
a'(\xi)(\psi, \hat{\varphi} + z) = 0 \quad \forall \psi \in X .
$$

Taking in this equation the test function $\psi := u - u_h$, gives the error presentation:

$$
c_{\mathrm{drag}} - c_{drag,h} = a'(\xi)(u - u_h, \varphi - z) \quad \forall \varphi \in X_h .
$$

As upper bound for the error we obtain:

$$
|c_{\mathrm{drag}} - c_{drag,h}| \le \|a'(\cdot)\| \, \|\nabla(u - u_h)\| \inf_{\varphi \in X_h} \|\nabla(z - \varphi)\| .
$$

This implies for Q_2-elements and $u, z \in H^3(\Omega)$ 4th-order convergence, i.e., for quasi-uniform meshes with maximum mesh size h and a constant c:

$$
|c_{\mathrm{drag}} - c_{drag,h}| \le c h^4 \|u\|_{H^3(\Omega)} \, \|z\|_{H^3(\Omega)} .
$$

4 Adaptivity

In this section, we present our approach for a posteriori error control for output functionals and quadratic finite elements. The basis is the standard approach of dual weighted residuals [7, 11] where a dual solution is used for computing weights entering the estimator. Here, we focus on the approximation of the interpolation error for quadratic elements and an algorithm how to adapt the mesh.

4.1 A posteriori error control for Q_2 elements

The aim is to get an a posteriori error estimator η for the discretization error with respect to functional output:

$$j(u) - j(u_h) \approx \eta \,.$$

The estimator makes use of the *continuous dual solution* $z \in X$ and the *discrete dual solution* $z_h \in X_h$:

$$\begin{aligned} a'(u_h)(z,\varphi) &= j'(u_h)(\varphi) \quad \forall \varphi \in X, \\ a'(u_h)(z_h,\varphi) &= j'(u_h)(\varphi) \quad \forall \varphi \in X_h \,. \end{aligned}$$

We recall the result of [7] for a scalar output functional j. By $i_h : X \to X_h$ we denote an arbitrary interpolation operator.

Theorem 1: *We have the error representation*

$$j(u) - j(u_h) = \frac{1}{2}\left\{\rho(u_h)(z - i_h z) + \rho^*(u_h, z_h)(u - i_h u)\right\} + R^* \tag{6}$$

with the primal and dual residuals

$$\begin{aligned} \rho(u_h)(\varphi) &:= (f,\varphi) - a(u_h)(\varphi)\,, \\ \rho^*(u_h,z_h)(\varphi) &:= j'(u_h)(\varphi) - a'(u_h)(\varphi, z_h)\,, \end{aligned}$$

and a remainder term R^* formally of third order with respect to the error $\{u - u_h, z - z_h\}$.

In the case of vector-valued output functionals, for each component of $j(u)$ an associated dual solution has to be used in this error representation. In order to use this result as mesh adaptation criterion we have to approximate the interpolation errors $\omega_z \approx z - i_h z$ and $\omega_u \approx u - i_h u$, and the resulting estimator has to be localized for giving information about the influence of the local mesh size h_K.

A cheaper variant of (6) has a remainder term of only second order but only needs the evaluation of the primal residual and the interpolation of the dual solution, see [11]:

Theorem 2: *We have the error representation*

$$j(u) - j(u_h) = \rho(u_h)(z - i_h z) + R \tag{7}$$

with a remainder term R formally of second order with respect to the error $\{u - u_h, z - z_h\}$.

For the numerical results given in the last section of this work, we only consider this cheaper variant (7). However, the interpolation technique to be presented applies directly to the representation (6).

After integration by parts the estimator reads:

$$\eta = \rho(u_h)(\omega_z) = \sum_{K \in \mathcal{T}_h} \left((f - Lu_h, \omega_z)_K + \frac{1}{2}\left(\mu \left[\frac{\partial v_h}{\partial n}\right] - pn, \omega_z \right)_{\partial K} \right) .$$

with a cell residual $(f - Lu_h, \omega_z)_K$ and a jump term across cell edges

$$\left[\frac{\partial v_h}{\partial n}\right](x) := \frac{\partial v_h(x)}{\partial n}\Big|_K - \frac{\partial v_h(x)}{\partial n}\Big|_{K'}, \tag{8}$$

for a point x on the edge $K \cap K'$. For quantitative error prediction, it is not necessary to compute both parts. It has been shown in Carstensen [12] that for linear finite elements the jump terms (8) dominate, whereas for quadratic elements the residual terms become dominant. Therefore, we take as cell error indicators the quantities

$$\eta_K := \|f - Lu_h\|_K \|\omega_z\|_K ,$$

giving the estimator

$$|j(u) - j(u_h)| \approx \sum_{K \in \mathcal{T}_h} \eta_K .$$

For the definition of the weights ω_z, we approximate the interpolation error $z - i_h z$ by discrete third-order difference quotients:

$$\|z - i_h z\|_K \approx C h_K^3 \|\nabla^3 z\|_K \approx C h_K^3 \|\nabla_h^3 z_h\|_K ,$$

with an interpolation constant C. Therefore, the weights ω_z are chosen on each cell K as

$$\omega_z|_K := C h_K^3 \|\nabla_h^3 z_h\|_K .$$

We still have to explain the definition of the discrete difference quotient ∇_h^3. For ease of presentation, we explain this procedure for a scalar quantity z. The extension to vector-valued quantities is straight forward. First, we approximate the Hessian of z_h by a matrix-valued discrete κ_z which is the L^2-projection of $\nabla^2 z_h$:

$$\kappa_z \in Q_h^{3\times 3} : \quad \sum_{K \in \mathcal{T}_h} (\nabla^2 z_h - \kappa_z, \phi)_K = 0 \quad \forall \phi \in Q_h^{3\times 3} .$$

Note, that $\nabla^2 z_h$ is discontinuous across cell edges. Second, the expression $\nabla_h^3 z_h$ is defined cell-wise by the gradients of κ_z:

$$\nabla_h^3 z_h|_K := (\partial_x + \partial_y + \partial_z)\kappa_z|_K .$$

Remark: Instead of taking for κ_z a L^2-projection one may take the gradient of another projection of $\nabla^2 z_h$, for instance, by taking the average nodal values of $(\nabla^2 z_h)|_K(\mathcal{N})$, for the nodes $\mathcal{N}$ of $\mathcal{T}_{h/2}$. This procedure is much cheaper because it is a local projection.

4.2 The mesh refinement strategy

In the previous section we described the localization strategy for obtaining error indicators η_K. Next we have to choose a subset of cells of the triangulation $\mathcal{T}_h$ for refinement. There are several standard approaches, see for instance [13]. Without loss of generality we assume $\eta_{K_i} \geq \eta_{K_{i+1}}$ for all $i \in \{1, \ldots, N\}$, where N stands for the number of cells of $\mathcal{T}_h$. The question of adaptive refinement can be reduced to the determination of a non-negative number m with $1 \leq m \leq N$ and a refinement of all cells K_i with $i \leq m$.

We denote by $\mathcal{T}_m$ the locally refined mesh resulting from $\mathcal{T}_h$ by refining these m cells. The corresponding number of cells in a $d-$dimensional domain becomes for isotropic bisections

$$N_m = N + m(2^d - 1) .$$

For the mesh $\mathcal{T}_m$ we consider the "global mesh size" $h \sim N_m^{-1/d}$. Furthermore, we denote by $\alpha > 0$ the largest number such that $j(u) - j(u_h) = O(h^\alpha)$ assuming optimal fitted meshes. For Q_2 discretization in this work we have $\alpha = 4$. Using an error estimator η_m as an approximation of the discretization error on $\mathcal{T}_m$ we obtain $\eta_m \sim N_m^{-\alpha/3}$. Now, the adaptive process aims at minimizing the constant

$$c_m = \frac{\eta_m}{N_m^{-\alpha/3}} ,$$

where we compute η_m as an extrapolation of the already computed estimate η for the triangulation $\mathcal{T}_h$ to the finer mesh $\mathcal{T}_m$:

$$\eta_m := \eta - \beta \sum_{i=1}^{m} \eta_{K_i} ,$$

with $\beta = 1 - 0.5^\alpha$. Now, the optimal choice for m can be easily determined by taking

$$m = \arg \min_{1 \le m \le N} c_m .$$

Remark: For a mesh $\mathcal{T}_h$ with equilibrated cell error indicators $\eta_K = \eta/N$ we have:

$$\begin{aligned} \eta_m &= \eta(1 - \beta m/N) , \\ c_m &= \eta(1 - \beta m/N) N_m^{\alpha/3} \sim (1 - \beta x)(1 + (2^d - 1)x)^{\alpha/3} , \end{aligned}$$

with $x := m/N$. For $d = 3$ and $\alpha = 4$, this number c_m has its minimum in $(0, 1]$ at $x = 1$. This results in $m = N$ and, therefore, to a global refinement. This is known to be optimal for equilibrated errors.

5 Results

In the following we document the results obtained by the adaptive process. For each configuration and each functional considered one adaptive run is performed. In total, we get six histories of local refinements. The stabilization constants are chosen for all runs of configuration 1 as $\alpha_0 = 0.05$ and $\delta_0 = 0.2$. For the second configuration with corner singularities (square cross-section), the values are chosen as $\alpha_0 = 0.1$ and $\delta_0 = 0.1$.

Because wrong local refinement may not converge or may converge to wrong values, we perform one step of global refinement after the final adaptive step. This makes the values obtained much more reliable. This is essential to guarantee the accuracy of the reference values. In the following tables, we indicate this last global refinement by a straight line in order to separate it to the adaptive local refinement. The computations on meshes with less than 2.5 million of degrees of freedom (dof) are performed on a single processor PC (AMD Opteron 246, 2 GHz) with 4 GB memory. Only on even finer meshes, a parallel computer (PC cluster) was necessary. In the following tables, also this is marked by a straight line and the suffix "serial" for computations on a PC and "parallel" computations on a cluster.

The nonlinear equations are solved by a Newton iteration. No pseudo-time stepping is necessary for the flows considered due to the relatively low Reynolds number of 20. The linear systems are solved with multigrid taking advantage of a hierarchy of (locally refined) meshes. The smoother is a block incomplete LU factorization, where the degrees of freedom of the velocities and the pressure are blocked together. For details we refer to [13, 14].

In the plots given in the following subsections we do not display the result on the finest (final) mesh, because this defines the reference values.

5.1 Circular cross-section

Pressure drop: We begin with the easier configuration with the circular cross-section. In Table 5, we list the values obtained of the pressure drop. The final step of a global refinement (increasing the number of cells by a factor of eight) shows that the local adaptation converges to the correct quantity. The first five leading digits of the pressure drop can be considered as correct: $j_1(u) = 0.171007 \pm 10^{-5}$.

In Figure 3, we compare with the results shown before in Table 2 on globally refined meshes. On a mesh with 1 383 Q_2 cells, a relative error of 1% with respect to the pressure drop is achieved. This accuracy is similar to the one on a globally refined mesh with more than 2.5 million cells. The computation on the locally refined mesh up to this accuracy needs 427 seconds on the single processor PC. This time includes the computation of the primal and dual problems up to the 4th mesh, the evaluation of the functional as well as the estimator and all I/O tasks. For a relative accuracy of 10^{-4}, only 14 375 cells are necessary. The corresponding CPU time is 4 881 seconds on the same machine again including the whole adaptive cycle with 8 primal and 8 dual solutions on subsequently refined meshes.

#cells	dof	Δp	machine
480	18 720	0.1892506787	
865	34 080	0.1798815121	
1 138	45 520	0.1746605518	
1 383	55 896	0.1729589594	
6 556	243 224	0.1712704796	
8 607	321 544	0.1710884110	
10 742	404 376	0.1710308434	
14 375	544 000	0.1709939769	
20 164	764 296	0.1710069896	
47 387	1 695 536	0.1710142741	
61 030	2 172 016	0.1710330777	serial
115 588	4 032 592	0.1710190472	parallel
924 704	32 260 736	0.1710070986	

Table 5: Values obtained for the circular cross-section on locally refined meshes with a posteriori error estimate of Δp.

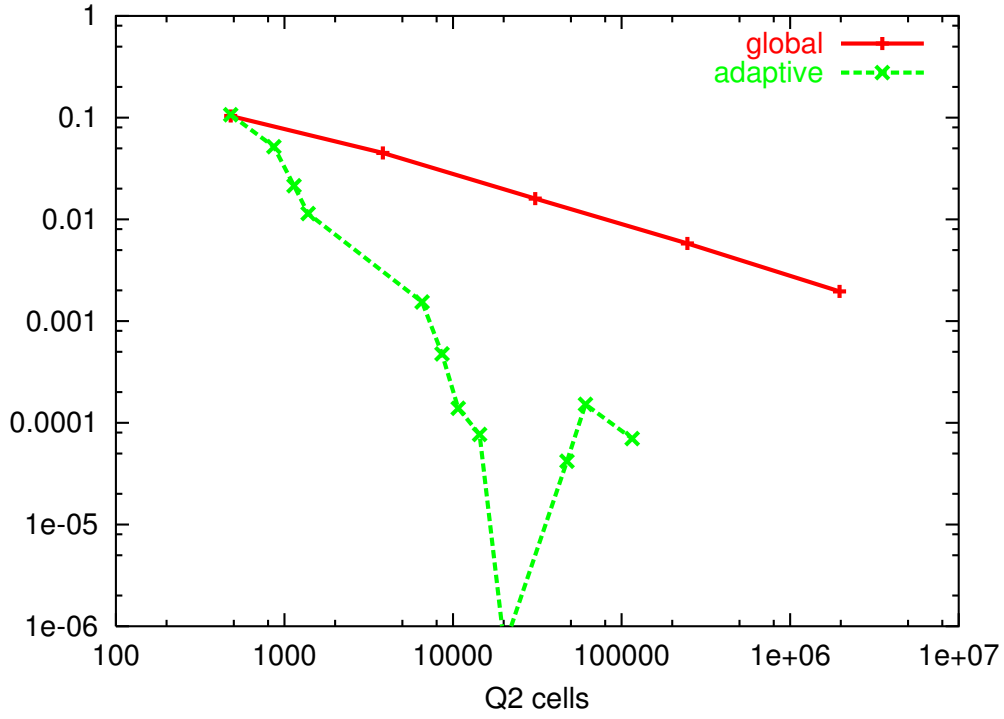

Figure 3: Circular cross-section: Comparison of global and local mesh refinement with respect to the relative errors in Δp.

Drag coefficient: For the drag and lift coefficient, the local refinement is not expected to give a gain that much because the primal solution u and the dual solution z are smooth. On globally refined meshes, 4th order convergence is obtained in [1] and with our discretization as well, see Table 2. However, as documented in Table 6 an enhancement of accuracy is observed. For instance, for a relative error of 10^{-6} a factor of 10 can be saved with local refinement, see Figure 4. But this is far beyond the relevant accuracy in practice. We can guarantee the first six to seven leading digits of the drag: $c_{\text{drag}} = 6.185333 \pm 10^{-7}$. The extrapolated value is obtained by assuming 4th order convergence in the last iteration.

#cells	dof	c_{drag}	machine
480	18 720	6.2503650	
3 840	136 000	6.17275342	
22 880	784 384	6.18471848	serial
155 768	5 227 872	6.18533571	parallel
1 246 144	41 822 976	6.18533310	
extrapolated		6.18533293	

Table 6: Values obtained for the circular cross-section on locally refined meshes with a posteriori error estimate of c_{drag}.

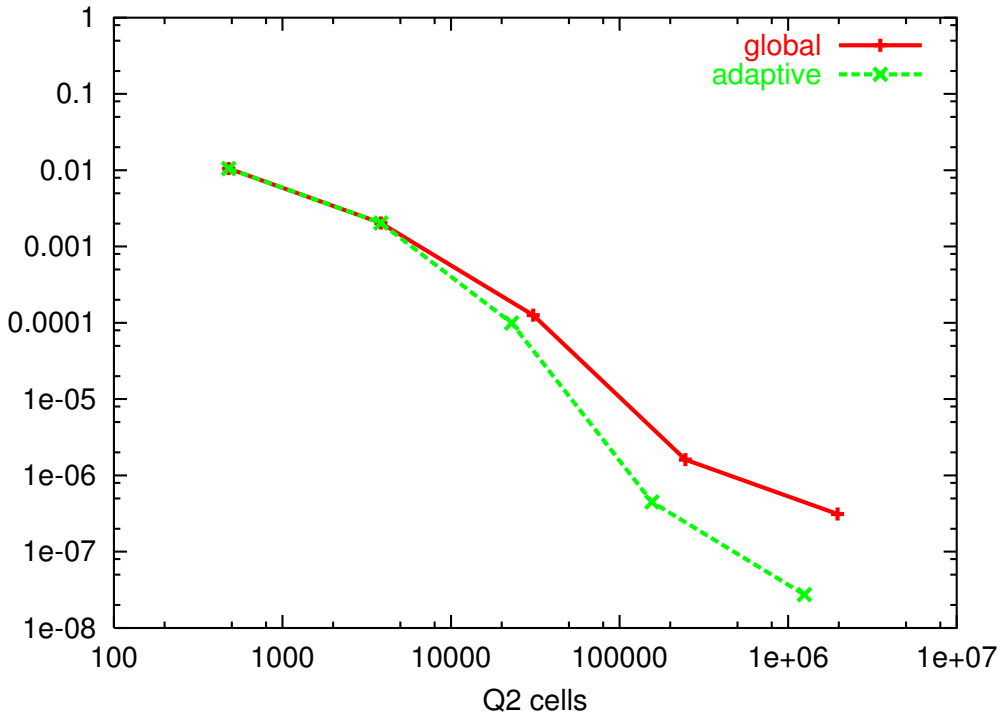

Figure 4: Circular cross-section: Comparison of global and local mesh refinement with respect to the relative errors in c_{drag}.

Lift coefficient: The lift coefficient is the most critical value in this configuration. From computations on globally refined meshes we only know the first three leading digits: $c_{\text{lift}} = 9.40 \cdot 10^{-3}$. Computations on locally refined meshes indicate that the next digit should be 1, so that the reference solution becomes $c_{\text{lift}} = 9.401 \cdot 10^{-3} \pm 3 \cdot 10^{-6}$. However, this last digit is not guaranteed. The plot in Figure 5 shows that the local refinement only gives slightly better results.

5.2 Square cross-section

For the geometry with corner singularities we expect an even better improvement by local mesh refinement, because neither the solution nor the dual one are smooth. The correct balance of resolving the singularities and the far field let us expect to be much more efficient than globally refined meshes.

Pressure drop: Comparing the pressure drop on the penultimate mesh with 146 896 cells with the one on the last mesh (one global refinement) shows that the change is about 10^{-5} and the reference solution can be considered as $\Delta p = 0.175686 \pm 5 \cdot 10^{-6}$. A comparison with the most accurate solution on globally refined meshes shows that we save a factor of more than 30 in the number of degrees of freedom. The comparison of the relative errors in Δp is shown in Figure 6.

#cells	dof	c_{lift}	machine
3 840	136 000	$1.023315808e^{-2}$	
21 536	740 416	$9.483555569e^{-3}$	serial
149 104	5 006 304	$9.405158225e^{-3}$	parallel
1 192 832	40 050 432	$9.401228291e^{-3}$	
extrapolated		$9.40097e^{-3}$	

Table 7: Values obtained for the circular cross-section on locally refined meshes with a posteriori error estimate of c_{lift}.

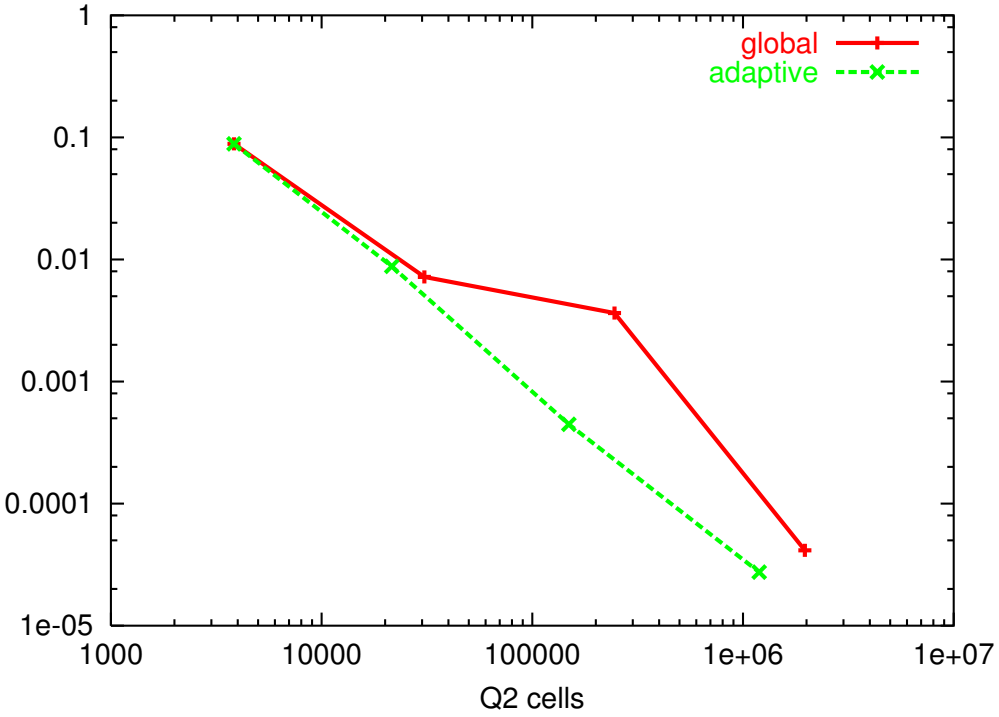

Figure 5: Circular cross-section: Comparison of global and local mesh refinement with respect to the relative errors in c_{lift}.

#cells	dof	Δp	machine
78	3 696	0.183495025	
624	24 544	0.208049746	
2 248	83 952	0.177348465	
5 160	191 424	0.176449284	
14 008	511 952	0.175792355	
33 496	1 212 208	0.175713985	serial
83 000	2 969 936	0.175676527	parallel
664 000	23 759 488	0.175686487	

Table 8: Values obtained in this work for the square cross-section on adaptively refined meshes for the pressure drop Δp.

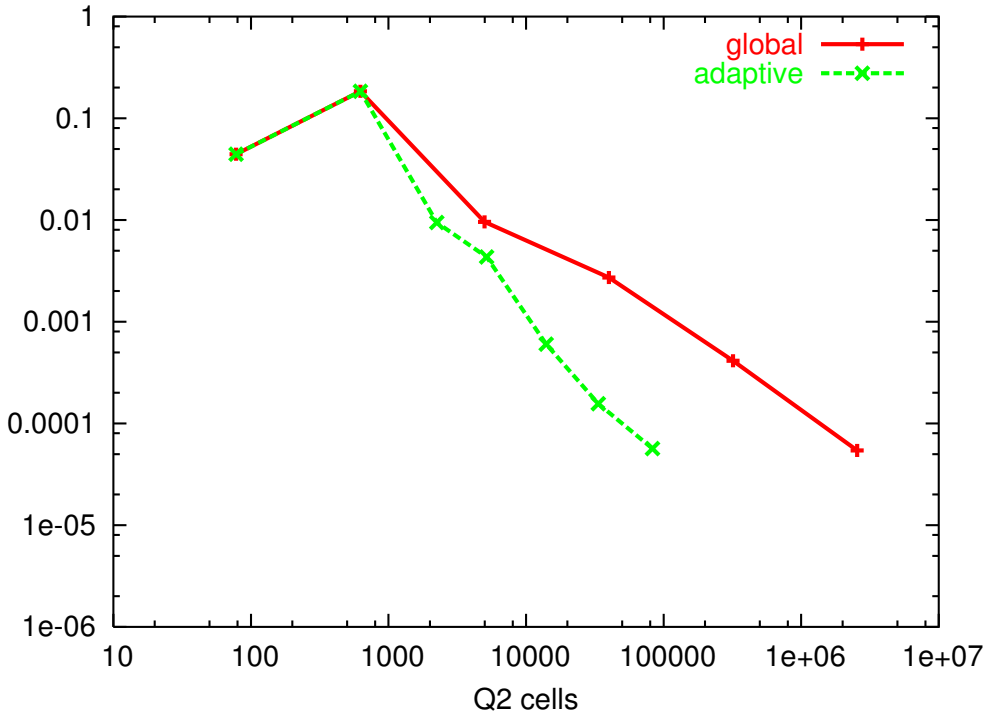

Figure 6: Square cross-section: Comparison of global and local mesh refinement with respect to the relative error in Δp.

Drag coefficient: The drag was determined on globally refined meshes only up to the first two leading digits (cf. Table 3). With local refinement we obtain: $c_{\text{drag}} = 7.767 \pm 2 \cdot 10^{-3}$. The plot in Figure 7 even shows a better asymptotic for the adaptive method. For 1% accuracy only 5 510 cells are needed instead of about 40 000 on a structured mesh (cf. Table 3).

#cells	dof	c_{drag}	machine
624	24 544	8.044496403	
2 472	91 120	7.974163476	
7 120	258 512	7.788092939	
16 808	615 408	7.759470239	
53 880	1 931 648	7.762036287	serial
113 128	4 052 272	7.765846534	parallel
905 024	32 418 176	7.767272368	

Table 9: Values obtained in this work for the square cross-section on adaptively refined meshes for the drag coefficient c_{drag}.

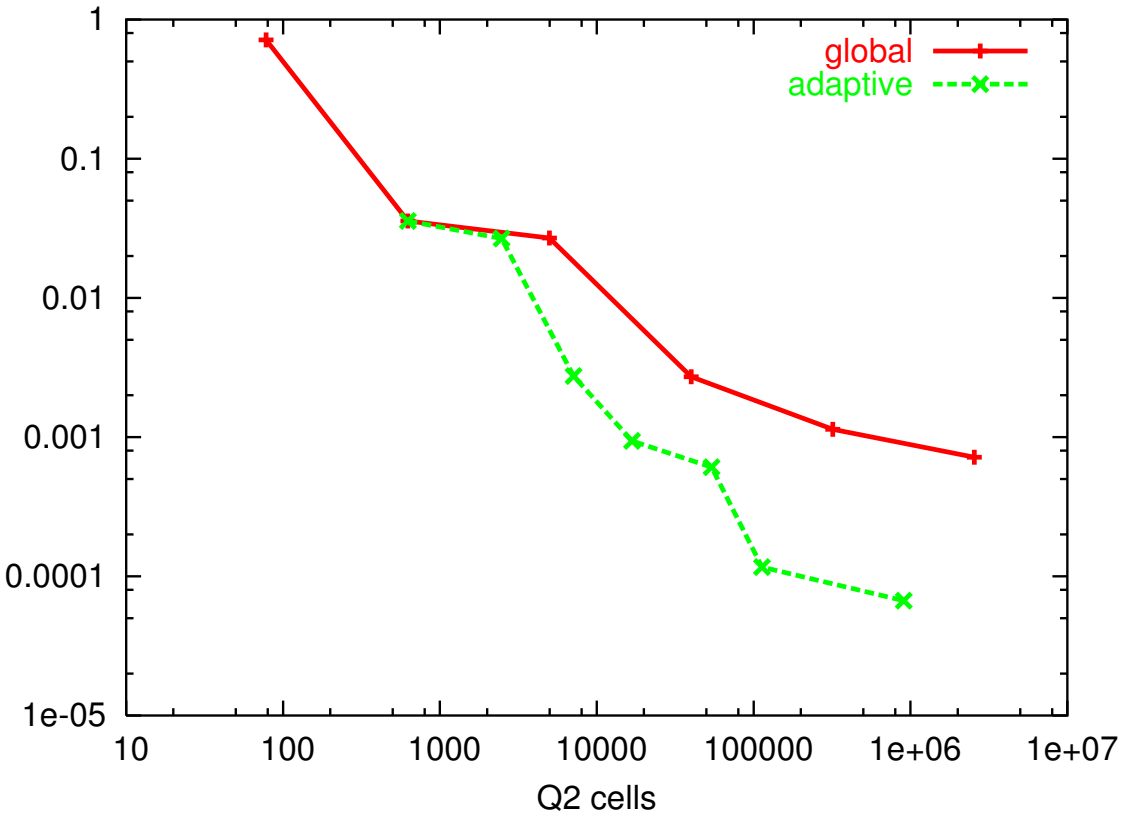

Figure 7: Square cross-section: Comparison of global and local mesh refinement with respect to the relative errors in c_{drag}.

Lift coefficient: Also in this case, the lift coefficient remains to be the most delicate quantity to compute. The first leading digits will be $c_{\text{lift}} = 6.893 \cdot 10^{-2} \pm 2 \cdot 10^{-2}$. The accuracy still profits from local refinement but the gain is not as much, see Figure 8.

#cells	dof	c_{lift}	machine
78	3 696	$5.880420044e^{-2}$	
435	17 608	$1.040065281e^{-1}$	
1 611	61 616	$7.794480009e^{-2}$	
5 643	210 336	$6.750582790e^{-2}$	
8 506	327 008	$6.818400747e^{-2}$	
56 764	2 037 464	$6.873510495e^{-2}$	serial
454 112	16 299 712	$6.892755977e^{-2}$	parallel

Table 10: Values obtained in this work for the square cross-section on adaptively refined meshes for the lift coefficient c_{lift}.

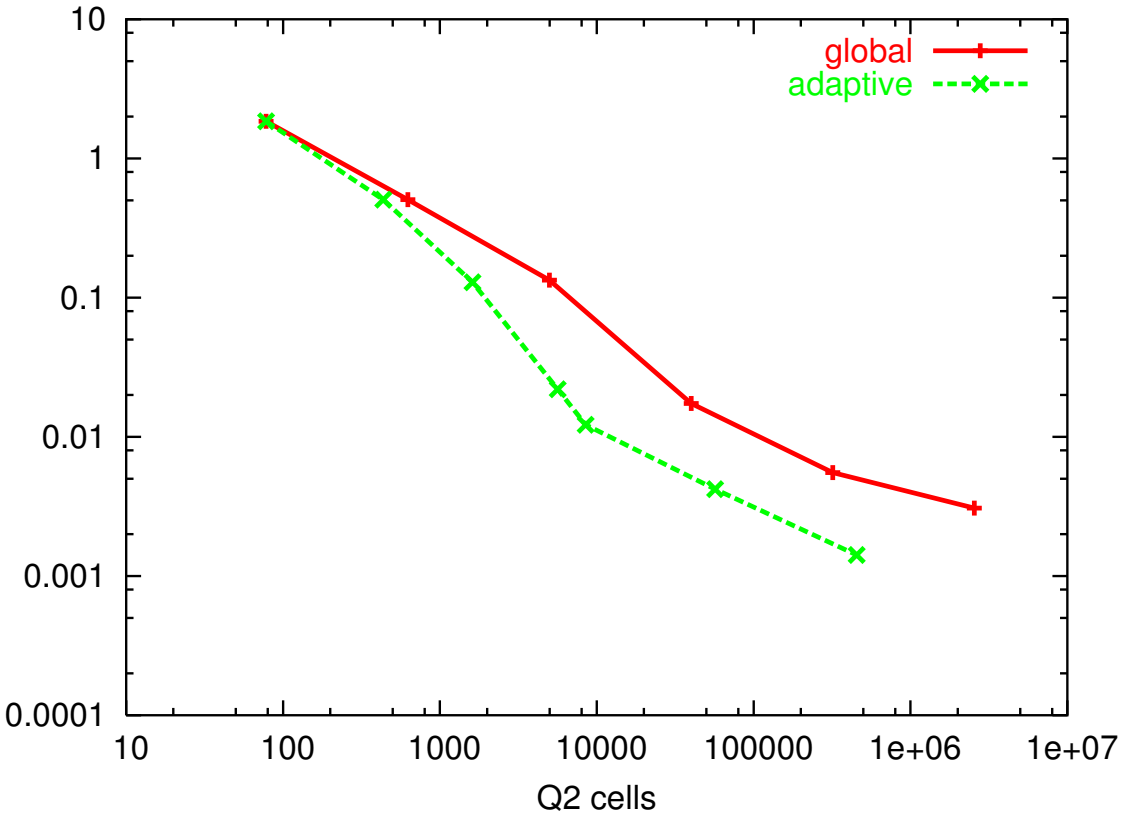

Figure 8: Square cross-section: Comparison of global and local mesh refinement with respect to the relative errors in c_{lift}.

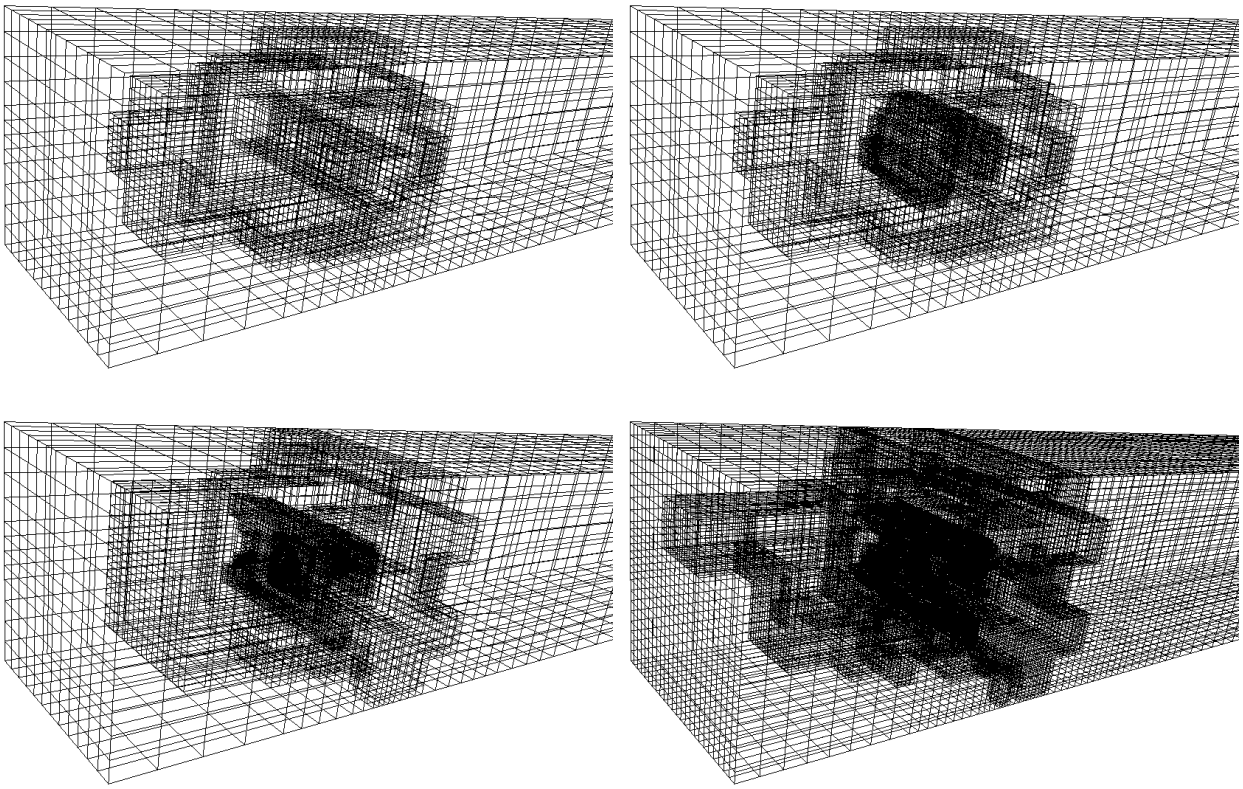

Figure 9: Zoom of four meshes obtained on the basis of a posteriori error control of the pressure drop.

Finally, we show in Figure 9 four meshes obtained within the adaptive mesh refinement for the pressure difference with increasing number of cells.

6 Summary

We applied a non-standard stabilized finite element method to still very hard Navier-Stokes benchmark problems in order to produce reliable reference values for functional output, as pressure drops, drag and lift coefficients. The accuracy of the method is substantially increased by applying a dual-weighted residual error estimator and local mesh refinement. It turns out, that we improve the reference values documented up to now in literature. In particular, we confirm the results in [1] and improve the accuracy of all quantities considered in that work. Furthermore, we document on reliable reference values for the even more difficult benchmark problem (geometry with corners).

In Table 11 we list the relative error obtained with the Q_2 discretization with local projection stabilization on a single PC for the two benchmark problems considered. We achieve, for instance, for the drag and for the pressure drop in the circular cross-section a relative accuracy of 10^{-4}. The lift remains the most delicate quantity. Only 1 % accuracy is obtained.

For higher accuracies with more than 2.5 million of degrees of freedom we used a parallelized version of the algorithm. The error in the reference values obtained are listed in Table 12. The best accuracy is obtained for the drag where at least 6 digits are computed accurately.

	Δp	c_{drag}	c_{lift}
circular	$< 10^{-4}$	$< 10^{-4}$	$< 10^{-2}$
square	$< 10^{-4}$	$< 10^{-3}$	$< 10^{-2}$

Table 11: Relative accuracy obtained on locally refined meshes on a single processor PC with locally refined meshes for the two configurations (circular and square).

	Δp	c_{drag}	c_{lift}
circular	$< 6 \cdot 10^{-5}$	$< 2 \cdot 10^{-8}$	$< 3 \cdot 10^{-4}$
square	$< 6 \cdot 10^{-5}$	$< 3 \cdot 10^{-4}$	$< 3 \cdot 10^{-3}$

Table 12: Relative accuracy obtained on locally refined meshes on a PC cluster with locally refined meshes for the two configurations (circular and square).

References

[1] V. John, Higher order finite element methods and multigrid solvers in a benchmark problem for 3D Navier-Stokes equations, Int. J. Numer. Math. Fluids. 40 (2002) 775–798.

[2] M. Schäfer, S. Turek, Benchmark computations of laminar flow around a cylinder. (With support by F. Durst, E. Krause and R. Rannacher), in: E. Hirschel (Ed.), Flow Simulation with High-Performance Computers II. DFG priority research program results 1993-1995, no. 52 in Notes Numer. Fluid Mech., Vieweg, Wiesbaden, 1996, pp. 547–566.

[3] R. Becker, M. Braack, A finite element pressure gradient stabilization for the Stokes equations based on local projections, Calcolo 38 (4) (2001) 173–199.

[4] R. Becker, M. Braack, A two-level stabilization scheme for the Navier-Stokes equations, in: e. a. M. Feistauer (Ed.), Numerical Mathematics and Advanced Applications, ENUMATH 2003, Springer, 2004, pp. 123–130.

[5] M. Braack, E. Burman, A multiscale method towards turbulent flow based on local projection stabilization, submitted to SIAM J. Numer. Anal. 2004.

[6] M. Braack, Solution of Complex Flow Problems: Mesh- and Model Adaptation with Stabilized Finite Elements, Habilitation thesis, Universität Heidelberg (2004).

[7] R. Becker, R. Rannacher, An optimal control approach to a posteriori error estimation in finite element methods, in: A. Iserles (Ed.), Acta Numerica 2001, Vol. 37, Cambridge University Press, 2001, pp. 1–225.

[8] P. Ciarlet, Finite Element Methods for Elliptic Problems, North-Holland, Amsterdam, 1978.

[9] C. Johnson, Numerical Solution of Partial Differential Equations by the Finite Element Method, Cambridge University Press, Cambridge, UK, 1987.

[10] J.-L. Guermond, Stabilization of Galerkin approximations of transport equations by subgrid modeling, M2AN 33 (6) (1999) 1293–1316.

[11] 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.

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

[13] M. Braack, An Adaptive Finite Element Method for Reactive Flow Problems, PhD Dissertation, Universität Heidelberg (1998).

[14] R. Becker, M. Braack, Multigrid techniques for finite elements on locally refined meshes, Numerical Linear Algebra with Applications 7 (2000) 363–379, special Issue.