Chapter 5

Optimization with Navier-Stokes equations

In this chapter, the optimization problem will be governed by the incompressible Navier-Stokes equations. General information about fluid mechanics and Navier-Stokes equations can be found for example in Temam [59], Ward-Smith [64] and White [65].

In Section 5.8 flow with temperature will be considered. The Boussinesq model will be used for the temperature.

In this chapter, Dirichlet boundary control (DBC) is applied. The developed methods and a posteriori error estimates show again good results. The new dual-weighted error estimator is more appropriate for grid design of optimization problems than the energy-error estimator. For the same target functional values, the dual-weighted error estimator needs less elements than the energy-error estimator. The refinement by the dual-weighted error estimator shows the sensitivities in the optimization problem. Another result is that the computations with the developed dual-weighted error estimator are more stable than with the energy-error estimator for the applications in this chapter.

Due to a new technique for the formulation and computation of (DBC), the regularization parameter α can be reduced considerably. This enables to get controls q which are not too strongly restricted by the given regularization profile. The resulting optimization problem is then less dominated by the regularization.

The coupled systems resulting from the applications in this chapter led to numerical problems. The solver developed in Section 6.1 enabled to solve the presented equation systems as described therein. It provided the necessary robustness. The preconditioner in section 6.2 led to a necessary acceleration of the solver.

Computations for both L^2 and H^1-regularization were performed as proposed in Gunzburger and Hou [34] (see section 1.8). The numerical results showed no considerable difference between the two versions in the test cases.

The globalization methods developed in section 3.3 have been used for some of the presented examples.

5.1 Navier-Stokes equations for flow simulation

The following version of the incompressible Navier-Stokes equations posed on a domain $\Omega \subset \mathbb{R}^d$, $d = 2$, in Cartesian coordinates are considered: First the two equations resulting from the momentum equation, leading to a conservation of momentum:

$$-\underbrace{\underbrace{\nu}_{\text{viscosity}} \quad \triangle v}_{\text{diffusion}} + \underbrace{v \cdot \nabla v}_{\text{transport}} + \nabla p = \vec{f}. \tag{5.1}$$

The two velocity components are denoted by $v = (u, w)$. The viscosity is denoted by ν, the pressure by p and $\vec{f} = (f_1, f_2)$ is the right hand side of the system. Let v be in $V \subset H^1(\Omega)^2$. And let p be in $T = L_2(\Omega)$. The weak formulation is

$$F_i = (v \cdot \nabla v, \psi)_\Omega + (\nu \nabla v, \nabla \psi)_\Omega - (p, \nabla.\psi)_\Omega - (\vec{f}, \psi)_\Omega + S_i = 0 \quad \forall \psi \in V,$$

with boundary conditions for $\partial\Omega = \Gamma_{in} \cup \Gamma_{out} \cup \Gamma_w$:

$$v|_{\Gamma_{in}} = v_{in}, \quad v|_{\Gamma_w} = 0, \quad (pn - \nu \partial_n v = 0)|_{\Gamma_{out}}.$$

Γ_w denotes a wall boundary condition. S_i is a suitable stabilization defined below.

The continuity equation, leading to conservation of mass, takes the form:

$$\nabla.v = 0. \tag{5.2}$$

The weak formulation is

$$F_c = (\nabla.v, \chi)_\Omega + S_c = 0 \qquad \forall \chi \in T. \tag{5.3}$$

S_c is a suitable stabilization defined below.

Since this formulation is not stable for the applied Q^1/Q^1-elements for the FEM discretization of v and p, the stabilization technique described in Becker [6] is used. The main difficulties for a stable discretization are the convection dominated behavior of the flow (small ν) and the velocity-pressure coupling. A standard finite element technique for the first difficulty is the streamline diffusion method, see also Johnson [42]. Both difficulties can be treated simultaneously through a so-called Galerkin-least-squares approach described in Becker [6, Chapters 5 and 6]. This techniques lead to the following stabilization: For the momentum equation we obtain

$$S_i = (v \cdot \nabla v + \nabla p, \tau \; v \cdot \nabla \psi)_\Omega.$$

And the continuity equation is stabilized by

$$S_c = (v \cdot \nabla v + \nabla p, \tau \; \nabla \chi)_\Omega.$$

The weighting parameter $\tau = \tau_K$ is calculated on each cell K of the finite element discretization. By [6] and [29], τ_K can be chosen approximately as $h_K \max(\frac{h_K}{\|\beta\|_{K,\infty}} - \nu, 0)$ leading to a stable FEM discretization. In this case, $\beta = \overline{u}$ can be a fixed velocity field, for example the one from a previous step of a fix point iteration. And h_K is the diameter of the cell K.

If $h_K \to 0$, then $\tau_K \to 0$. The LBB condition of the stabilized system

$$\inf_{r \in T_h} \sup_{v \in V_h} \left\{ \frac{\int_\Omega r \ \nabla \cdot v \ dx}{\|r\|_{L^2} \|\nabla v\|_{H^1}} + \frac{\int_\Omega c(r,r)}{\|r\|_{L^2}} \right\} \ \geq \ \gamma_h \ \geq \ \gamma \ > \ 0$$

with weighted Laplacian $c(p,r) = \tau(\nabla p, \nabla r)$ leads to a stable calculability of the pressure for the Stokes equation.

5.2 Stokes and Navier-Stokes - Poiseuille flow

As first test case for the equations derived in section 5.1, simulation calculations for Poiseuille flow are done on the domain presented in Figure Fig 1.1 in chapter 1 (see White [65, section 3-3.1, p. 116-118] and Ward-Smith [64, section C1, p. 135-137]). For Poiseuille flow, the transport term $v \cdot \nabla v$ disappears because the flow is parallel to the axis. Therefore, there is no difference between Stokes and Navier-Stokes equations.

For test purposes, the numerical results obtained with the code 'of' show that the implemented system leads to the expected solutions. In the following table, the values of the residuals $R(.)$ are given for certain numbers N of cells. The results are from the computation with the Stokes equations.

N	$R(u)$	$R(w)$	$R(p)$
256	$2.6 * 10^{-12}$	$3.7 * 10^{-11}$	$6.5 * 10^{-11}$
1024	$5.4 * 10^{-14}$	$5.7 * 10^{-13}$	$6.6 * 10^{-13}$
4096	$1.2 * 10^{-15}$	$1.6 * 10^{-14}$	$1.4 * 10^{-14}$
16384	$1.5 * 10^{-15}$	$1.4 * 10^{-14}$	$5.3 * 10^{-15}$

5.3 Bifurcation for pure simulation

For the case of computation on the domain on page 90, the following problem could be stated for the pure simulation: For a Reynolds number smaller than 70, a convergence to a stationary solution can be stated. At Reynolds number 80, the solution is non-stationary. But for Reynolds number 90 and 100, again the former stationary solution is observed.

It is well-known that for small data there is a stationary solution of the mathematical model Navier-Stokes equations. A detailed survey can be found in Heywood, Rannacher and Turek [37]. But from Reynolds number 70, an additional non-stationary solution exists. This is the solution which can be stated for some Reynolds numbers between 70 and 80. Here bifurcations can happen, i.e. both solutions can occur as solutions of the Navier-Stokes equations. For Reynolds number 90 and 100, the stationary 'base' solution was obtained which is physically unstable.

5.4 Lagrangian function and its differentials

In this section, the equation system for optimization with the incompressible Navier-Stokes equations as application is derived. The presented terms are only those terms which result directly from the incompressible Navier-Stokes equations. Terms resulting from other parts of the optimization system like the cost functional, control terms or other boundary conditions must be added in correspondence with the special optimization problem.

The Lagrangian function for Cartesian coordinates can be stated as usual:

$$L(v,p,q,\lambda,\lambda_2) := J(v,p,q) + (\lambda_2, F_c(v,p,q)) + (\lambda, F_i(c,p,q)). \tag{5.4}$$

The derived equations do not contain stabilization. A detailed version of this equation system with stabilization is given in appendix B.

The first order necessary conditions of the constraint optimization problem are for the weak formulation of the Navier-Stokes equations omitting possible boundary integrals from the optimization problem:

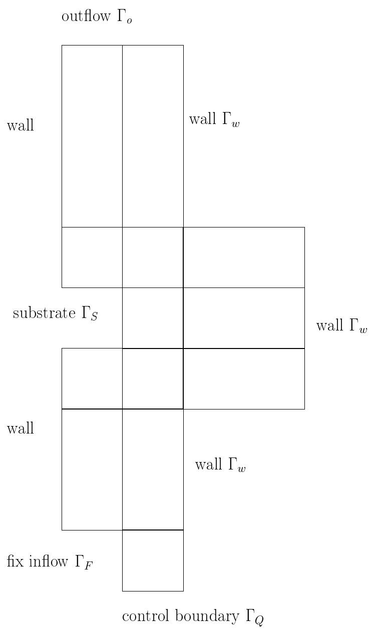
$$\frac{\partial L}{\partial v} = (\lambda \cdot \nabla v - v \cdot \nabla \lambda, \psi)_\Omega - (\nu \nabla \lambda, \nabla \psi)_\Omega - (\nabla.\lambda_2, \chi)_\Omega = 0 \quad \forall \psi \quad \forall \chi \tag{5.5}$$

$$\frac{\partial L}{\partial p} = -(\lambda, \nabla.\psi) = 0 \quad \forall \psi \tag{5.6}$$

The differentials $\frac{\partial L}{\partial \lambda}$ and $\frac{\partial L}{\partial \lambda_2}$ lead to the state equations $F_i = 0$ and $F_c = 0$, respectively. The possible boundary integrals from the optimization problem depend on the special problem and will therefore be considered separately for each presented optimization problem. The second-order differentials are obtained as in Section 4.2 by second-order differentiation in the direction of the increments (see remarks on equation (1.10)). The detailed Hessian operator is given in appendix B. The differentials of L with stabilization are much bigger due to the nonlinearities and products in the stabilization S_i and S_c.

5.5 A drag coefficient optimization problem

The following basic grid will be used for the presented drag optimization problem of this section:



The drag coefficient c_D is calculated in the following way:

$$c_D = \frac{2F_D}{\rho \overline{U}^2 L}$$

with fluid density ρ, mean velocity $\overline{U}$, 'length' L of substrate Γ_S and the drag force F_D as follows:

$$F_D = \int_S (\rho\nu \frac{\partial}{\partial n} v_t n_y - P n_x) dS$$

with normal vector n on S with x-component n_x and y-component n_y, tangential velocity v_t on S and tangent vector $t = (n_y, -n_x)$. There are two possibilities for the below given domain: The boundary line which is the border of S is perpendicular to the flow direction. Then,

$$v_t = u \;\Rightarrow\; \frac{\partial}{\partial n} v_t = \partial_y u.$$

Otherwise, boundary line which is the border of S is in the flow direction. Then,

$$v_t = w \;\Rightarrow\; \frac{\partial}{\partial n} v_t = \partial_x w.$$

For the calculated problem, the following values were used: The kinematic viscosity $\nu = 10^{-3}\ \frac{\mathrm{m}^2}{\mathrm{s}}$ and the fluid density $\rho = 1\ \frac{\mathrm{kg}}{\mathrm{m}^3}$.

A first test case is to take simply $J(u,q) = c_D$ as cost functional. This lead to an ill-posed problem because c_D is not lower bounded. Therefore, condition (H1) in Gunzburger and Hou [34], section 2.2. on page 1003, is not fulfilled. The numerical results showed this problem. The control function diverged with minimal value $\to -\infty$, as expected.

The optimization problem with

$$J(u,q) = c_D^2$$

as cost functional leads to better results. By [34], section 2.2. on page 1003, this optimization problem is well-posed because the conditions (H1)-(H9) are fulfilled. For this cost functional J, the theoretical minimum is $c_D^2 = 0$. This value is taken as reference value J_{ref} in the computations and results of this section. It results from $J_{ref} = 0$ that $E_h = J(u_h, q_h) - J_{ref} = J(u_h, q_h)$.

OPTDRAG: The calculations were done on the following domain or basic grid: There is one fixed inflow Γ_F (bottom, at left). Next to it is the control boundary Γ_Q (bottom, at right). The drag is evaluated on the boundary of the substrate Γ_S in the middle. The outflow Γ_o is at the top.

The following boundary conditions are considered: On the fixed inflow Γ_F there is $u = 0,\ w(x,y) = -4(x - \frac{1}{2})^2 + 1$ (parabolic profile). On the control boundary Γ_Q there is $u = 0,\ w = q$ (DBC). On the wall Γ_w there is $u = w = \lambda_u = \lambda_w = 0$. At the outflow Γ_o there is the free outflow condition for u, w proposed in Becker [6] which is also transformed to λ_u, λ_w. The substrate Γ_S takes the same boundary conditions as the wall Γ_w.

Remark 5.5.1. *For a Reynolds number greater than 46 on the domain on page 90, the above equation system for Navier-Stokes leads to oscillations in the solution with the developed solution method. In order to get a more stable solution for the Navier-Stokes equations, the following technique is applied: One reason for problems in computation of solutions for Navier-Stokes equations results from Newton linearization of the transport term $v \cdot \nabla v$. This term can is split in two terms, the convection term and the reaction term. The reaction term $v \nabla \overline{u}$, where $\overline{u}$ means that u is fixed for this iteration (or the value from the previous iteration), is the main source for problems. This term can cause problems e.g. for vortices. It is neglected in the equation system.*

5.6 Dual-weighted a posteriori error estimates

The a priori error estimates for optimization problems governed by incompressible Navier-Stokes equations can be found in Gunzburger and Hou [34, theorem 4.10].

For the case of pure simulation, a posteriori error estimates for incompressible Navier-Stokes equations are given in Becker [6, theorem 4.2].

The following version of the dual-weighted a posteriori error estimate for optimization problems is a general version for the domain integrals ($T \in \mathbf{T}_h$). The integrals on $\partial\Omega$ and on Γ_Q depend on the special optimization problem (and will be evident from the formulation of each optimization problem). Later on, the cost functional will be evaluated on a part of the domain. This will change slightly the error estimator.

Proposition 5.6.1. *For optimization problems governed by incompressible Navier-Stokes equations, there holds the following dual-weighted* a posteriori *error estimate for (DBC) (on the boundary Γ_Q) and cost functional $\min J = c_D^2$ with drag coefficient c_D on the boundary Γ_S as in* **OPTDRAG***:*

$$\begin{aligned}
|J(u,q) - J(u_h,q_h)| \;\le\;& \sum_{\Gamma\subset\partial\Omega} h_\Gamma^2 \{\rho_\Gamma^{(\lambda^{(2)})}\omega_\Gamma^{(\lambda^{(2)})} + \rho_\Gamma^{(w)}\omega_\Gamma^{(w)}\} + \sum_{\Gamma\subset\Gamma_Q} h_\Gamma^2 \rho_\Gamma^{(q)}\omega_\Gamma^{(q)} \\
&+ \sum_{\Gamma\subset\partial\Omega} h_\Gamma^2 \rho_\Gamma^{(p)}\omega_\Gamma^{(p)} \qquad (5.7)\\
&+ \sum_{K\in\mathbf{T}_h} h_K^2 \{\rho_{\partial K}^{(u)}\omega_{\partial K}^{(\lambda^{(0)})} + \rho_{\partial K}^{(w)}\omega_{\partial K}^{(\lambda^{(1)})} + \rho_{\partial K}^{(p)}\omega_{\partial K}^{(\lambda^{(2)})}\} \\
&+ \sum_{K\in\mathbf{T}_h} h_K^2 \{\rho_{\partial K}^{(\lambda^{(0)})}\omega_{\partial K}^{(u)} + \rho_{\partial K}^{(\lambda^{(1)})}\omega_{\partial K}^{(w)} + \rho_{\partial K}^{(\lambda^{(2)})}\omega_{\partial K}^{(p)}\},
\end{aligned}$$

with the cell residuals and weights

$$\begin{aligned}
\rho_\Gamma^{(\lambda^{(2)})} &= h_\Gamma^{-3/2}\|2c_D \cdot n_x\|_\Gamma, \quad \Gamma\subset\Gamma_S, & \omega_\Gamma^{(\lambda^{(2)})} &= h_\Gamma^{-1/2}\|p - \chi_h^{(p)}\|_\Gamma, \\
\rho_\Gamma^{(w)} &= h_\Gamma^{-3/2}\|w_h - q_h\|_\Gamma, \quad \Gamma\subset\Gamma_Q, & \omega_\Gamma^{(w)} &= h_\Gamma^{-1/2}\|\lambda^{(1)} - \chi_h^{(w)}\|_\Gamma, \\
\rho_\Gamma^{(q)} &= h_\Gamma^{-3/2}\|\partial_n\lambda_h^{(1)} - \alpha q_h\|_\Gamma, & \omega_\Gamma^{(q)} &= h_\Gamma^{-1/2}\|q - \chi_h^{(q)}\|_\Gamma, \\
\rho_\Gamma^{(p)} &= h_\Gamma^{-3/2}\|\partial_n p_h\|_\Gamma, & \omega_\Gamma^{(p)} &= h_\Gamma^{-1/2}\|\lambda^{(2)} - \zeta_h^{(p)}\|_\Gamma,
\end{aligned}$$

$$\begin{aligned}
\rho_{\partial K}^{(u)} &= \tfrac{1}{2}h_K^{-3/2}\|n\cdot[\nabla u_h]\|_{\partial K\setminus\partial\Omega}, & \omega_{\partial K}^{(\lambda^{(0)})} &= h_K^{-1/2}\|\lambda^{(0)} - \pi_h^{(u)}\|_{\partial K\setminus\partial\Omega},\\
\rho_{\partial K}^{(w)} &= \tfrac{1}{2}h_K^{-3/2}\|n\cdot[\nabla w_h]\|_{\partial K\setminus\partial\Omega}, & \omega_{\partial K}^{(\lambda^{(1)})} &= h_K^{-1/2}\|\lambda^{(1)} - \pi_h^{(w)}\|_{\partial K\setminus\partial\Omega},\\
\rho_{\partial K}^{(p)} &= \tfrac{1}{2}h_K^{-3/2}\|n\cdot[p_h]\|_{\partial K\setminus\partial\Omega}, & \omega_{\partial K}^{(\lambda^{(2)})} &= h_K^{-1/2}\|\lambda^{(2)} - \pi_h^{(p)}\|_{\partial K\setminus\partial\Omega},\\
\rho_{\partial K}^{(\lambda^{(0)})} &= \tfrac{1}{2}h_K^{-3/2}\|n\cdot[\nabla\lambda_h^{(0)}]\|_{\partial K\setminus\partial\Omega}, & \omega_{\partial K}^{(u)} &= h_K^{-1/2}\|u - \psi_h^{(u)}\|_{\partial K\setminus\partial\Omega},\\
\rho_{\partial K}^{(\lambda^{(1)})} &= \tfrac{1}{2}h_K^{-3/2}\|n\cdot[\nabla\lambda_h^{(1)}]\|_{\partial K\setminus\partial\Omega}, & \omega_{\partial K}^{(w)} &= h_K^{-1/2}\|w - \psi_h^{(w)}\|_{\partial K\setminus\partial\Omega},\\
\rho_{\partial K}^{(\lambda^{(2)})} &= \tfrac{1}{2}h_K^{-3/2}\|n\cdot[\lambda_h^{(2)}]\|_{\partial K\setminus\partial\Omega}, & \omega_{\partial K}^{(p)} &= h_K^{-1/2}\|p - \psi_h^{(p)}\|_{\partial K\setminus\partial\Omega}.
\end{aligned}$$

$(\psi_h^{(u)},\psi_h^{(w)},\psi_h^{(p)}),(\pi_h^{(u)},\pi_h^{(w)},\pi_h^{(p)}),(\chi_h^{(u)},\chi_h^{(w)},\chi_h^{(p)},\zeta_h^{(p)})$ are arbitrary test functions in the discrete spaces. The cell residuals obtained from the first order necessary conditions are omitted because the jump terms will dominate the residual terms ([13]).

Proof. See theorem 2.5.1, and proposition 2.5.2. □

The energy error estimator is built as in Section 2.7. In this application we have several state equations.

$$\eta_E(u_h) := c_I \sum_{K\in\mathbf{T}_h} h_K^3(\rho_{\partial K}^{(u)2} + \rho_{\partial K}^{(w)2} + \rho_{\partial K}^{(p)2}) + c_I \sum_{\Gamma\subset\partial\Omega} h_\Gamma^3(\rho_\Gamma^{(u)2} + \rho_\Gamma^{(w)2} + \rho_\Gamma^{(p)2}), \quad (5.8)$$

with the cell residuals $\rho_{\partial K}^{(.)}$ and $\rho_\Gamma^{(.)}$.

5.7 Numerical results for the drag coefficient optimization problem

There are 2 possibilities to achieve the value $c_D^2 = 0$ in optimization problem **OPTDRAG**: One is that all inflow on Γ_F is sucked off by an outflow on the control boundary Γ_C (negative inflow). The result is that there is no flow near Γ_S and $c_D^2 = 0$. This solution is obtained by the optimization problem governed by the Stokes equation in Figure 5.2. The other possibility for $c_D^2 = 0$ is that the drag on Γ_S is 0 by mutual elimination of the non-zero parts. This enables to have flow near Γ_S. It forces a change of direction in the flow. This solution is obtained by the optimization problem governed by the Navier-Stokes equation in Figure 5.8. The difference in the solutions results from the additional nonlinear transport term in the Navier-Stokes equations.

The cost functional

$$J(e) = E_h = c_D^2$$

contains the square of the drag coefficient c_D. This means that in the presented results, the drag coefficient values in the optimization problems can be retrieved by taking the square root of J.

Remark 5.7.1. *The value of the efficiency index has with this setting at least to weaknesses: Some residual terms in the error estimator are neglected. And the regularization is not added to the cost functional. The latter has the advantage that J gives the value of c_D.*

The numerical results are obtained either with the pure Newton method or with the modified Newton method.

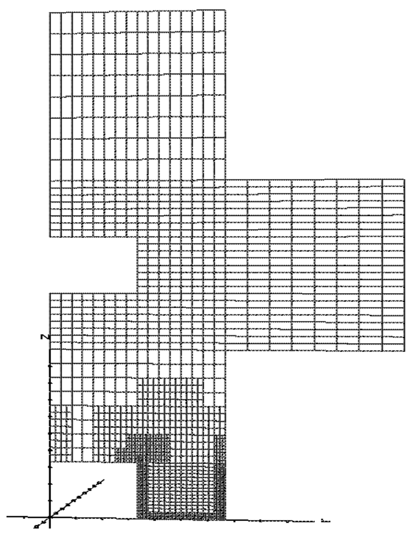
Example 1: First, this optimization problem is governed by the *Stokes equations.* The starting values for $(u, w, p), (\lambda_u, \lambda_w, \lambda_p)$ and q are $(0.2, -0.3, 2), (0, 0, 0.1)$ and -0.3, respectively. The calculations were done with the dual-weighted and the energy-error estimator. The regularization parameter α is 0.01. Divergence was detected with α lower than 10^{-8}. Without the (DBC)-technique of section 6.6 α has to be taken greater than 80 in order to obtain convergence. The interpolation constant is $C_I = 0.1$. The following table shows the value of the error functional E_h and the efficiency index I_{eff} for certain discrete solutions with N cells of the results obtained with the dual-weighted error estimator:

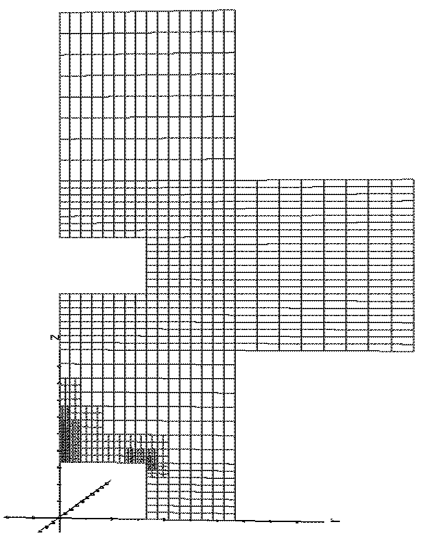
N	964	2020
E_h	$1.2 * 10^{-5}$	$1.59 * 10^{-6}$
I_{eff}	1.26	0.41

For these discrete solutions, the parts of the dual-weighted error estimator have the following values in the notation of proposition 5.6.1:

N	964	2020
$\sum \rho_\Gamma \omega_\Gamma, \Gamma \subset \Gamma_S$	$8.1 * 10^{-6}$	$1.3 * 10^{-6}$
$\sum \rho_\Gamma \omega_\Gamma, \Gamma \subset \Gamma_Q$	$9 * 10^{-9}$	$6 * 10^{-8}$
$\sum \rho_{\partial T} \omega_{\partial T}$	$1.4 * 10^{-6}$	$3.8 * 10^{-5}$

In the error estimator given in proposition 5.6.1, many terms from the domain integral are neglected. This leads to the observed values of the efficiency index I_{eff}. Nevertheless, the generated grids are quite efficient as shown in the comparison with the energy-error estimator. In Figure 5.1, a faster convergence to the solution of the underlying continuous problem can be stated. This is due to the more adequate refinement for the optimization problem by the dual-weighted error estimator as presented in the following grids. They show that the energy-error estimator focuses mainly on the fix inflow where the gradient has great values. The obtained grids by dual-weighted (left) $[c_D^2 = 1.588 * 10^{-6}]$ and energy (right) $[c_D^2 = 0.000145284]$ error estimator both on 2000 elements for $\alpha = 0.01$ are





The drag value $1.59*10^{-6}$ for the dual-weighted error estimate shows that the theoretical minimum $c_D^2 = 0$ can almost be achieved with the dual-weighted error estimator. In this linear case for the simulation, the control tries to eliminate the flow next to the observation which is the evaluation of the drag. This is one possibility to force the drag to be close to 0. An other possibility is achieved with the Navier-Stokes equations in the following examples.

Remark 5.7.2. *Small changes in the control q can lead to remarkable gains in the cost functional J. So for some applications, the difference in the controls may not to visible because it is very small. On coarser grids the difference of the obtained controls are often more visible because the dual-weighted error estimator leads faster to the optimal control than the energy-error estimator. Furthermore not only the control is important but also to obtain a grid which enables to get a certain accuracy of J with the least number of cells.*

In the following Figure, we observe remarkable differences in the controls due to the fact that the mesh refinement by the energy-error estimator is near the fix inflow boundary Γ_F and not near the control boundary Γ_Q. Therefore, the energy-error estimator does not lead to the optimal control obtained by the dual-weighted error estimator. This is a reason for the worse values for c_D^2 on the grids designed by the energy-error estimator (see Figure 5.1).

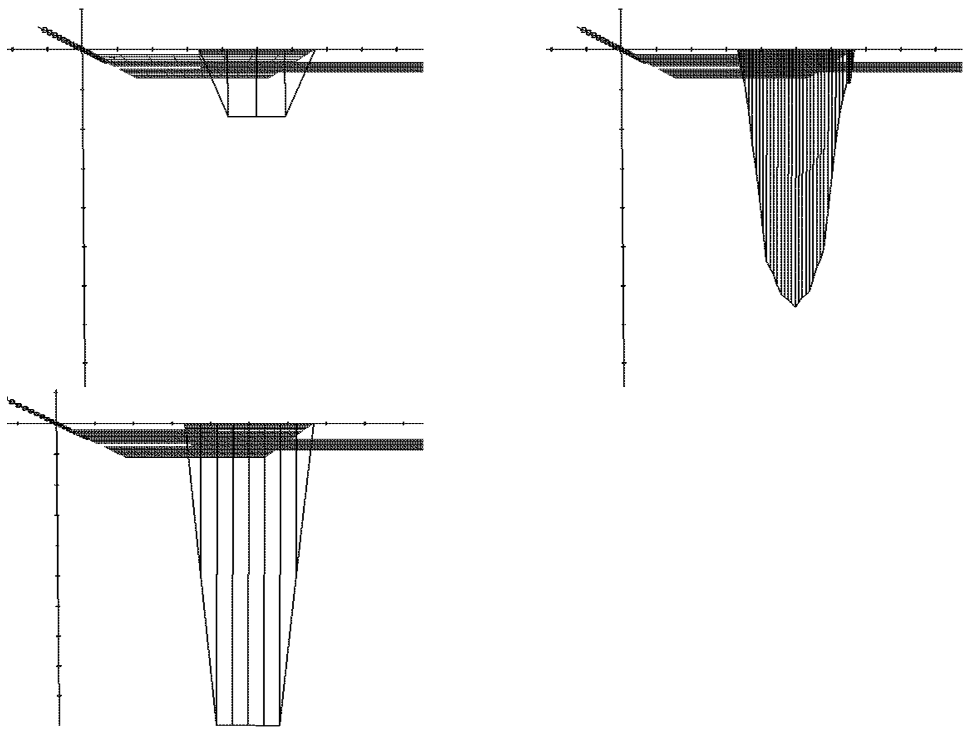


Figure: Drag optimization problem governed by Stokes equations: Obtained controls by dual-weighted (above) and energy (below) error estimator on N=200, 2000, and 2300 elements, with minimal values = -0.3, -1.17, -0.3, respectively, for α = 0.01 (note the different scaling of the first and third control).

Example 2: The next step is the optimization problem governed by the *Navier-Stokes equations*. The data is the same as for the calculations with the Stokes equations. Numerical results for $\nu^{-1} = 15, 20, 70$ will be presented. The presented results for E_h in Figures 5.3, 5.4, 5.5, 5.6 and 5.7 show that the dual-weighted error estimator is much more appropriate for grid design of optimization problems. The energy-error estimator does not really consider the optimization problem and therefore the resulting refinement is inefficient. For example, in Figure 5.5 the cost function value 0.000167 can be obtained with the dual-weighted error estimator on 7000 elements whereas the energy-error estimator needs 66000 elements.

The following tables show the value of the error functional E_h and the efficiency index I_{eff} for certain discrete solutions with N cells of the results obtained with the dual-weighted error estimator. For $\nu^{-1} = 20$, $\alpha = 0.01$ we get:

N	832	9076	92812
E_h	0.00043345	$8.305 * 10^{-5}$	$1.647 * 10^{-5}$
I_{eff}	1.2	0.99	0.8

For $\nu^{-1} = 70$, $\alpha = 0.007$ we obtain:

N	2044	9652	48376
E_h	0.000353775	$6.592 * 10^{-5}$	$1.132 * 10^{-5}$

In the latter table, the values of I_{eff} are omitted because the showed bad values. Here, remark 5.7.1 may be applicable. In the case $\nu^{-1} = 70$, the effect of the neglected residual terms are clearly observed by the resulting numerical values. Due to the greater Reynolds number, the flow may have a stronger influence on the whole equation system. For $\nu^{-1} = 20$, the values of the dual-weighted error estimator are close to the error.

The notation of the terms in the following two tables is as in proposition 5.6.1. For these discrete solutions, the parts of the dual-weighted error estimator have the following values for $\nu^{-1} = 20$, $\alpha = 0.01$:

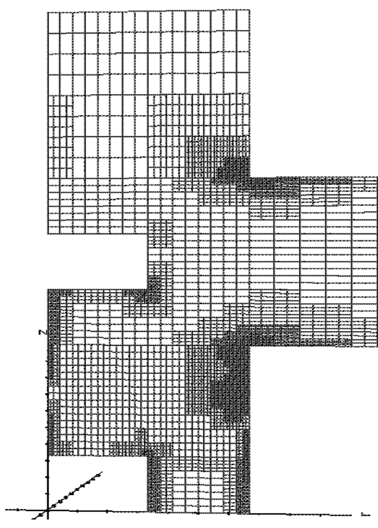
N	832	9076	92812
$\sum \rho_\Gamma \omega_\Gamma, \Gamma \subset \Gamma_S$	$1.921 * 10^{-6}$	$1.736 * 10^{-7}$	$1.1 * 10^{-9}$
$\sum \rho_\Gamma \omega_\Gamma, \Gamma \subset \Gamma_Q$	$1.42 * 10^{-7}$	$1.27 * 10^{-8}$	$2.9 * 10^{-9}$
$\sum \rho_{\partial T} \omega_{\partial T}$	0.00034254	$8.3381 * 10^{-5}$	$2.0525 * 10^{-5}$

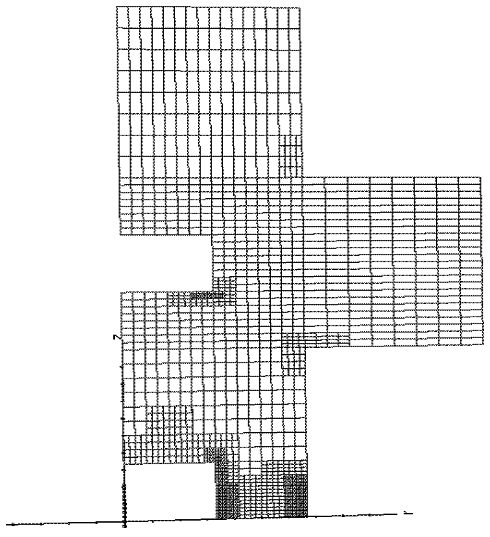
And for $\nu^{-1} = 70$, $\alpha = 0.007$ we get:

N	2044	9652	48376
$\sum \rho_\Gamma \omega_\Gamma, \Gamma \subset \Gamma_S$	$7.086 * 10^{-7}$	$3.552 * 10^{-7}$	$7.3 * 10^{-9}$
$\sum \rho_\Gamma \omega_\Gamma, \Gamma \subset \Gamma_Q$	$4.064 * 10^{-8}$	$2.53523 * 10^{-7}$	$5.213 * 10^{-8}$
$\sum \rho_{\partial T} \omega_{\partial T}$	0.00386682	0.00574349	0.00250141

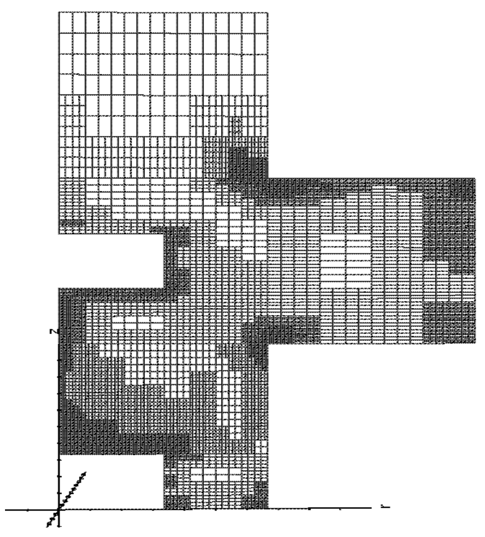
For $\nu^{-1} = 70$, the error estimator is dominated by $\rho_{\partial T} \omega_{\partial T}$. The values are greater as for the case $\nu^{-1} = 20$. This leads to the bad values for I_{eff} in combination with the neglected residual terms.

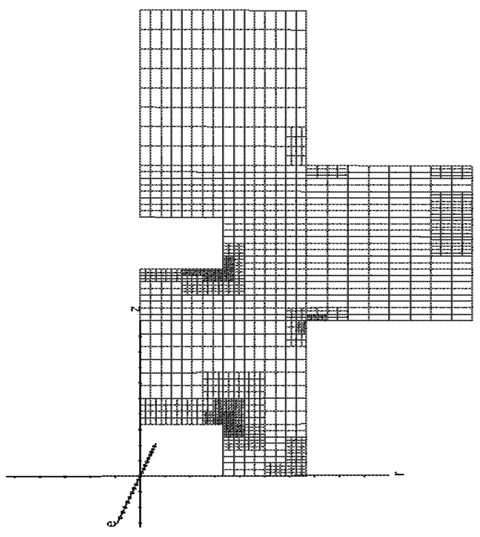
The behavior of the dual-weighted error estimator and the energy-error estimator for the different versions of the drag optimization problem governed by Navier-Stokes equations can be seen in the resulting grids. The obtained grids by dual-weighted (left) [$c_D^2 = 8.305*10^{-5}$, 10000 elements] and energy (right) [$c_D^2 = 0.00034999$, 20000 elements] error estimator for $\nu^{-1} = 20$ and $\alpha = 0.01$ are



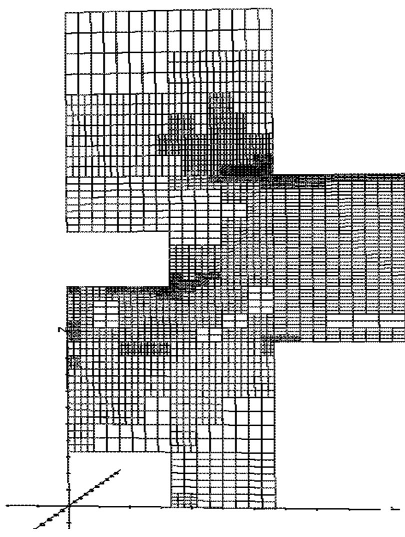


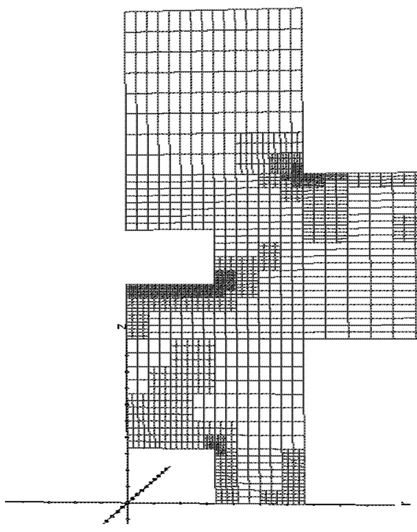
Whereas the obtained grids by dual-weighted (left) [$c_D^2 = 6.488 * 10^{-5}$, 14000 elements] and energy (right) [$c_D^2 = 0.00031092$, 10000 elements] error estimator for $\nu^{-1} = 15$ and $\alpha = 0.007$ are





The obtained grids by dual-weighted (left) [$c_D^2 = 8.051*10^{-5}$, 6000 elements] and energy (right) [$c_D^2 = 0.00036091$, 11000 elements] error estimator for $\nu^{-1} = 70$ and $\alpha = 0.007$ are





Sequences showing the grid design for several discretization levels for the dual-weighted and the energy-error estimator can be found in figures 5.11 and 5.12, respectively.

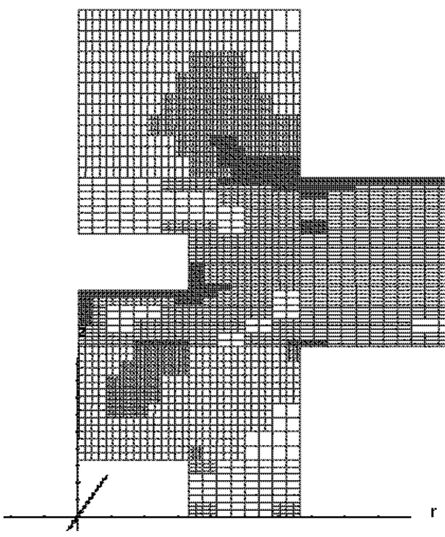
For fine grids, the differences in the controls are very small for this application (see Remark 5.7.2). The control can be obtained from the Figures for the velocity by the Dirichlet boundary condition $u = q$ on Γ_Q.

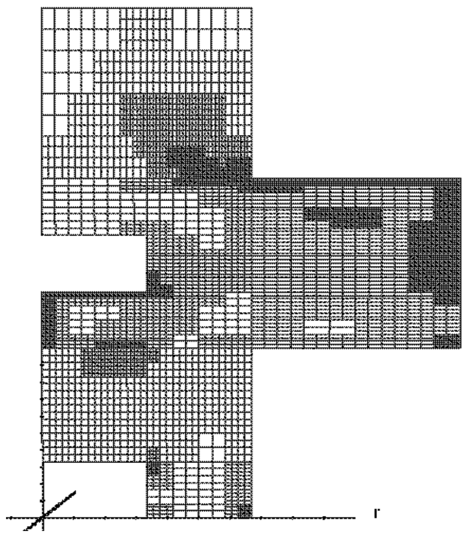
The mesh refinement in the middle part of the grid by the dual-weighted error estimator in the presented grids for $\nu^{-1} = 15$, 20 and 70 can be explained by the necessity of change of the flow direction in order to obtain $c_D^2 = 0$. This leads to a sensitivity to the evaluation of the cost functional.

The energy-error estimator leads to a refinement in mainly by the corner singularities (grids for $\nu^{-1} = 15$) or by the flow (grids for $\nu^{-1} = 70$).

For greater Reynolds numbers, the impact of the flow on the optimization problems is getting stronger. This can also be seen in the resulting grids for $\nu^{-1} = 70$.

The grids clearly show that the regularization has an impact on the grid design. Larger values for α lead to a too strong dominance of the regularization which can be seen also in the mesh refinement. This can be seen by comparison of grids for $\nu^{-1} = 20$ and grids for $\nu^{-1} = 15$ for which different values of α were used. The choice of an appropriate α is crucial for a good solution of the optimization problem. If α is too big, the obtained solution is dominated by the regularization. The resulting solution is in this case not the solution sought in the original optimization problem. The extreme case would be that there will be a control $q \equiv 0$ for very big values of α. But if α is too small, the well-definedness of the optimization problem may be lost and no good convergence of the solution method would result. For comparison, the following two grids are the same configuration as for the last grid for the dual-weighted error estimator. Again, $\nu^{-1} = 70$ in the drag optimization problem. But the regularization factor α is 0.0085 (left, $c_D^2 = 8.8 \cdot 10^{-5}$) and 0.01 (right, $c_D^2 = 10^{-4}$) both with 6000 cells as above:





Clearly, the effect of the increasing regularization factors α not only on the region next to the control boundary can be seen.

The velocities for flow with fixed control $q = 0$ is given in Figure 5.9 for the case of $\nu^{-1} = 70$ and $\alpha = 0.007$. This solution corresponds to the solutions in Figure 5.8. It shows the effect of the control on the flow. A comparison of the belonging numerical solutions of the pressure is shown in Figure 5.10. Again, the impact of the control due to the sucking off on the control boundary Γ_Q is obvious.

The largest computation done with the Navier-Stokes equations in this application was with 2 million variables for the solutions obtained with the dual-weighted error estimator. With the energy-error estimator such large amounts of variables were not obtained because the calculations were much slower. This results from inefficient refinement and a numerically less stable behavior.

For all presented examples, the cells representing 30 percent of the error were refined in each refinement step (starting with the largest error values).

5.8 Optimization governed by flow with temperature transport for zero gravitation

The next application of the developed techniques is optimization governed by Navier-Stokes equations with temperature. For the temperature, the Boussinesq viscosity model in White [65, p. 482] and Griebel, Dornseifer, and Neunhoeffer [32, p. 125] is used. The gravitation is neglected. Hence the temperature is used as a tracer. The solution of the following equation represents the temperature component:

$$\epsilon(\nabla T, \nabla \phi) + (v \cdot \nabla T, \phi) = 0. \tag{5.9}$$

For the associated Lagrangian multiplier λ_t is determined by the equation

$$\epsilon(\nabla \lambda_T, \nabla \phi) - (v \cdot \nabla \lambda_T, \phi) = 0. \tag{5.10}$$

For stability reasons, only the terms

$$(v \cdot \nabla \delta T, \phi), (v \cdot \nabla \delta \lambda_T, \phi) \tag{5.11}$$

are taken in the second order derivation. This technique is similarly already described in remark 5.5.1.

The stabilization is facilitated by the term

$$\tau_T (v \cdot \nabla T, v \cdot \nabla \phi). \tag{5.12}$$

The factor τ_t can be chosen in at least two ways. The first possibility only includes information of the velocity:

$$\tau_T = \frac{1}{2} \frac{h}{|v|_{L^2}}. \tag{5.13}$$

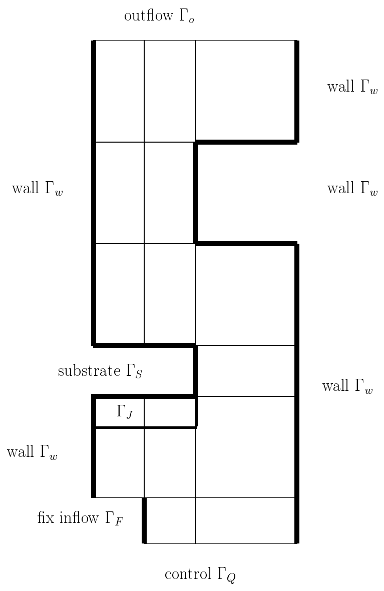
The second possibility takes more information and is close to the τ of the Navier-Stokes equations:

$$\tau_T = \frac{1}{70 * \frac{\nu}{h^2} + \frac{|v|_{L^2}}{h}}. \tag{5.14}$$

The presented results are achieved with the second version of τ_T.

All numerical results are obtained with the pure Newton method.

OPTTEMP: For this application, the optimization problem is as follows: The temperature T in the region Γ_J next to the substrate Γ_S should be maximized. The calculations were done on the following domain or basic grid:



There is one fixed inflow Γ_F (bottom, at left). Next to it is the control boundary Γ_Q (bottom, at right). The 'outflow' Γ_o is at the top.

The following boundary conditions are considered: On the fixed inflow Γ_F there is $u = 0$, $w(x,y) = -4(x-\frac{1}{2})^2 + 1$ (parabolic profile). On the control boundary Γ_Q there is $u = 0$, $w = q$ (DBC). On the wall Γ_w there is $u = w = \lambda_u = \lambda_w = 0$. At the outflow Γ_o there is the free outflow condition for u, w proposed in Becker [6] which is also transformed to λ_u, λ_w. The substrate Γ_S takes the same boundary conditions as the wall Γ_w. The boundary conditions for the solution of the Boussinesq equation temperature T are Dirichlet boundary conditions on all boundaries. These Dirichlet values for T are the same on all boundaries with exception of the boundary Γ_F. On boundary Γ_F there is a greater value than on the other boundaries. Therefore, there are Dirichlet boundary conditions with value 0 for λ_T on all boundaries.

This optimization problem is close to the optimization problem of the application mentioned in the introduction.

The dual-weighted error estimates are as follows for **OPTTEMP**: *For optimization problems governed by incompressible Navier-Stokes equations and Boussinesq temperature equation, there holds the following dual-weighted* a posteriori *error estimate for (DBC) (on the boundary Γ_Q) and cost functional* $\max J = \int_{\Gamma_J} T dx$ *as in* **OPTTEMP**: *In the notation of proposition 5.6.1,*

$$\begin{aligned} |J(u,q) \quad - \quad & J(u_h,q_h)| \le \sum_{\Gamma\subset\partial\Omega} h_\Gamma^2 \rho_\Gamma^{(w)}\,\omega_\Gamma^{(w)} + \sum_{\Gamma\subset\Gamma_Q} h_\Gamma^2\rho_\Gamma^{(q)}\,\omega_\Gamma^{(q)} \\ + \quad & \sum_{\Gamma\subset\partial\Omega} h_\Gamma^2\rho_\Gamma^{(p)}\,\omega_\Gamma^{(p)} + \sum_{\Gamma\subset\Gamma_J} h_\Gamma^2\rho_\Gamma^{(T)}\,\omega_\Gamma^{(T)} \\ + \quad & \sum_{K\in\mathbf{T}_h} h_K^2\{\rho_{\partial K}^{(u)}\,\omega_{\partial K}^{(\lambda^{(0)})} + \rho_{\partial K}^{(w)}\,\omega_{\partial K}^{(\lambda^{(1)})} + \rho_{\partial K}^{(p)}\,\omega_{\partial K}^{(\lambda^{(2)})} + \rho_{\partial K}^{(T)}\,\omega_{\partial K}^{(\lambda^{(3)})}\} \\ + \quad & \sum_{K\in\mathbf{T}_h} h_K^2\{\rho_{\partial K}^{(\lambda^{(0)})}\,\omega_{\partial K}^{(u)} + \rho_{\partial K}^{(\lambda^{(1)})}\,\omega_{\partial K}^{(w)} + \rho_{\partial K}^{(\lambda^{(2)})}\,\omega_{\partial K}^{(p)} + \rho_{\partial K}^{(\lambda^{(3)})}\,\omega_{\partial K}^{(T)}\}, \end{aligned} \tag{5.15}$$

with the additional cell residuals and weights

$$\rho_\Gamma^{(T)} = \tfrac{1}{2}h_K^{-3/2}\|n\|_\Gamma, \quad \Gamma\subset\Gamma_J\subset\Omega, \qquad \omega_\Gamma^{(\lambda^{(3)})} = h_K^{-1/2}\|\lambda^{(3)} - \pi_h^{(T)}\|_\Gamma, \tag{5.16}$$

$$\rho_{\partial K}^{(T)} = \tfrac{1}{2}h_K^{-3/2}\|n\cdot[T_h]\|_{\partial K\backslash\partial\Omega}, \quad \omega_{\partial K}^{(\lambda^{(3)})} = h_K^{-1/2}\|\lambda^{(3)} - \pi_h^{(T)}\|_{\partial K\backslash\partial\Omega}, \tag{5.17}$$

$$\rho_{\partial K}^{(\lambda^{(3)})} = \tfrac{1}{2}h_K^{-3/2}\|n\cdot[\lambda_h^{(3)}]\|_{\partial K\backslash\partial\Omega}, \qquad \omega_{\partial K}^{(T)} = h_K^{-1/2}\|T - \psi_h^{(T)}\|_{\partial K\backslash\partial\Omega}. \tag{5.18}$$

In addition to proposition 5.6.1, $\psi_h^{(T)}, \pi_h^{(T)}$ are appropriate test functions.

For the presented optimization problem, obtained discrete numerical solutions are shown in Figure 5.13. The control of the velocity on Γ_Q leads to a sucking off. So the original outflow gets to an inflow. The reason is that with this flow the temperature in the region Γ_J can be maximized as formulated by the optimization problem. Due to the change of the direction of the velocity, the temperature is not sucked off by the flow but is concentrated in the region Γ_J. There is then more higher temperature staying in the region Γ_J. The strange shape of the pressure p is due to the sucking off by the computed control on Γ_Q. The temperature T is as expected for this flow. The associated Lagrangian multiplier λ_T shows the searched duality to T and fits very well in the developed concept of error estimation.

On fine grids, the difference in the controls on the grids obtained by the dual-weighted error estimator and the energy-error estimator are not visible for this application (see remark 5.7.2). In the following Figure, a difference can be stated because the grids are still coarse. The belonging values of J are in Figures 5.17 and 5.18.

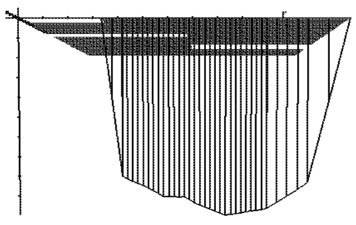
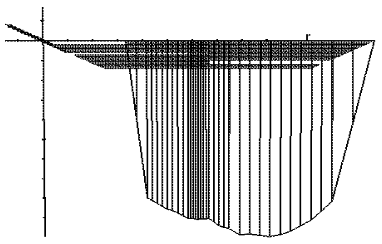
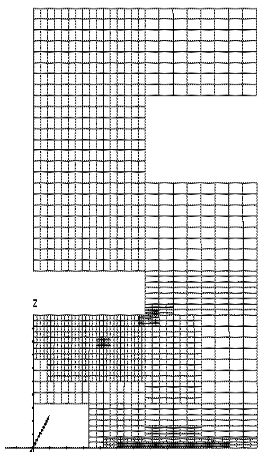
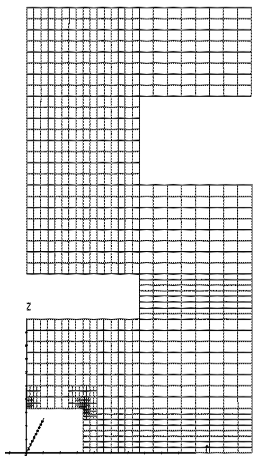
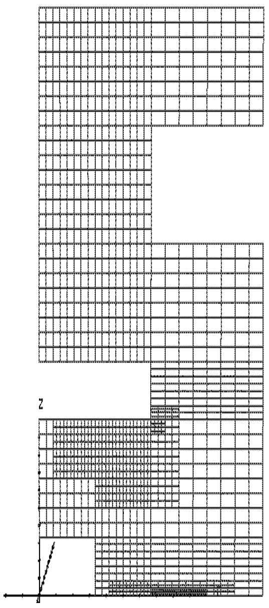



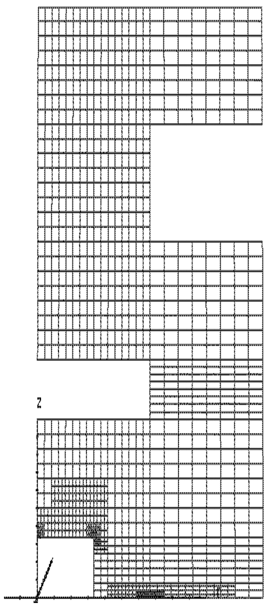
Figure: Temperature optimization problem governed by Boussinesq flow: Obtained controls by dual-weighted (left) and energy (right) error estimator on N=1300 and 1500 cells, with minimal values = -0.098 and -0.087, respectively, for $\alpha = 0.01$ and $\kappa = 0.5$.

In the following, a comparison of the grids obtained by the dual-weighted and the energy-error estimator is given. All computations are done with $\alpha = 0.01$ and pure Newton method. The grids depend on the value κ of the error which should be reduced in each refinement (applying the fixed fraction strategy, see section 2.8). $\kappa = 1$ means uniform refinement. First, the obtained grids by dual-weighted (left) [$J = 306.75$, 1580 elements] and energy (right) [$J = 70.19$, 1340 elements] error estimator for $\alpha = 0.01$ and $\kappa = 0.3$ with pure Newton method are presented. The energy error estimator is rather unstable because refinement occurs only near the inflow at corner singularities and steep gradients due to the 'inflow'. For the energy-error estimator divergence starts on 1700 cells. The dual-weighted error estimator behaves much more stable. Convergence up to 6700 cells was stated.

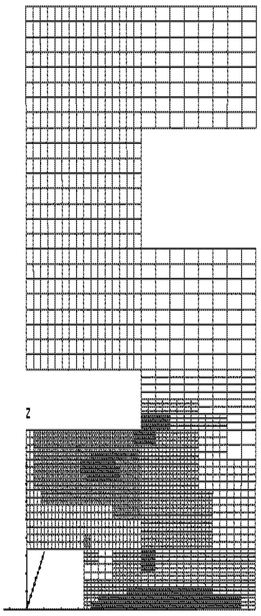



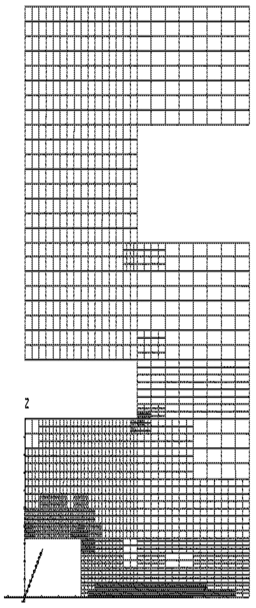
With $\kappa = 0.5$, the computations are also not very stable. The obtained grids by dual-weighted (left) [$J = 256.74$, 1300 elements] and energy (right) [$J = 70.31$, 1500 elements] error estimator are



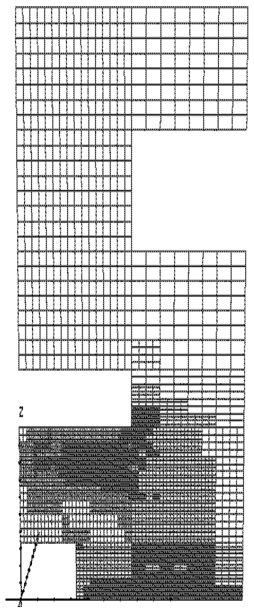


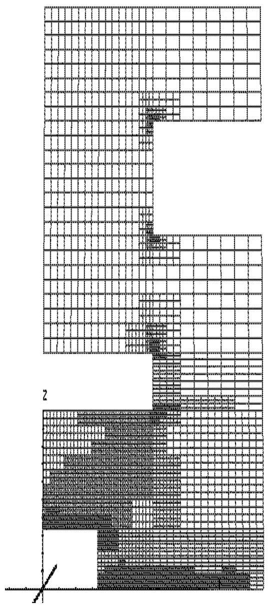
With $\kappa = 0.8$, the energy-error estimator behaves more stable. Computations up to 3800 cells are possible with the energy-error estimator. The dual-weighted error estimator reached 190000 cells. The obtained grids by dual-weighted (left) [$J = 1559.91$, 5000 elements] and energy (right) [$J = 282.38$, 3800 elements] error estimator are



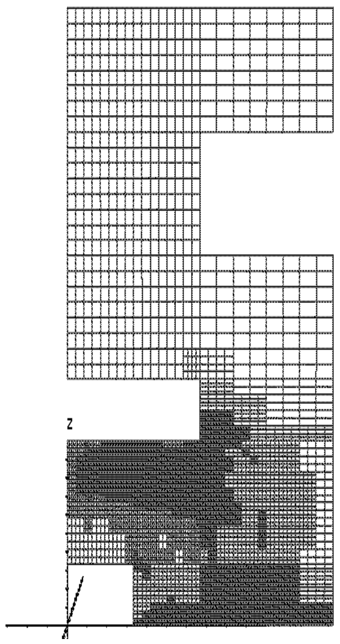


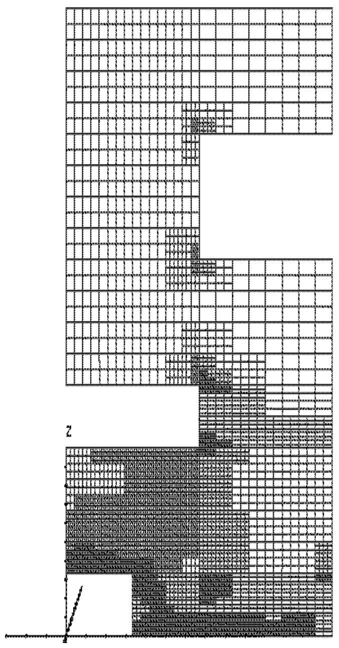
For $\kappa = 0.9$, even computations on 8300 cells can be done in a stable way with the energy-error estimator. For the the grids designed by the dual-weighted error estimator, 204000 cells were reached without any divergence. Afterwards, the computations were too big for the used work stations. The obtained grids by dual-weighted (left) [$J = 2600$, 7700 elements] and energy (right) [$J = 1007.83$, 8300 elements] error estimator are





For $\kappa = 0.95$, stable computations on 15800 cells are possible with the energy-error estimator. For the the grids designed by the dual-weighted error estimator, the computations showed no divergence. The computations were too big for the used work stations. The obtained grids by dual-weighted (left) [$J = 4016.97$, 10000 elements] and energy (right) [$J = 893.76$, 6500 elements] error estimator are





The computations with grids designed by the energy-error estimator were all much more unstable than those designed by the dual-weighted error estimator. In all cases, computations with much more cells were possible with the dual-weighted error estimator.

The energy-error estimator tends to refinement at corner singularities and steep gradients due to the 'inflow' on Γ_F and Γ_Q. Whereas the dual-weighted error estimator considers the optimization problem in its refinement. The designed grids show the sensitivities of the optimization problem. Therefore, the computed values for the cost functional to be maximized are higher with the obtained discrete solution compared to the cost functional values of the energy-error estimator grids with the same number of cells.

The value in the first iteration of the cost functional J was 17.46 on 224 cells (in all cases on a uniform refined basic grid). The obtained maximal value for J was 74776.57 (by the dual-weighted error estimator with $\kappa = 0.9$ on 204572 cells leading to 1.7 million variables).

In this optimization problem, the temperature in the region Γ_J has to be maximized. The main criterion for the evaluation of the error estimation and the thereby designed grids is therefore the (maximization of the) value of the cost functional J. In this spirit, the calculus of orientation at the value for E_h has to be replaced by the maximization of the value of J. For $\kappa = 0.95$, the values of the cost functional J in the limit of the Newton iteration on each discrete level for the dual-weighted and energy-error estimator are shown in Figure 5.14. All lines are plotted up to divergence of the computation or the line ends at the box of the graphics. The values of J are clearly greater for the grids designed by the dual-weighted error estimator. Compared to the uniform mesh refinement, the dual-weighted error estimator enables to reduce the number of cells by a factor of 6 in order to obtain the same value of the cost functional J. This behavior is also stated by the results obtained with $\kappa = 0.9$ and 0.8 in Figures 5.15 and 5.16, respectively.

Figure 5.17 shows a comparison of the values obtained with the energy-error estimator. The strong dependence of divergence and κ is especially significant compared with the dual-weighted error estimator in Figure 5.18. Again, all lines are plotted up to divergence of the computation or the line ends at the box of the graphics (besides mesh refinement which got too big i.e. for 'dual_weighted0.95'). For all considered κ, the computations with the dual-weighted error estimator were far more stable than those with the energy-error estimator. The energy error estimator places many vertices at the wrong regions of the domain. Furthermore these vertices are concentrated in a few regions by the energy-error estimator. This makes the computations more unstable as for the dual-weighted error estimator.

The different error values in the successful dual-weighted error estimator can be seen in the following tables. For $\kappa = 0.95$, these values are in the notation of proposition 5.6.1:

N	1448	32684	111572
$\rho_\Gamma \omega_\Gamma, \Gamma \subset \Gamma_Q$	$8.87062 * 10^{-6}$	$2.79328 * 10^{-7}$	$5.59577 * 10^{-8}$
$\rho_{\partial K} \omega_{\partial K}$	0.404636	0.0123389	0.00388003

For $\kappa = 0.9$, these values are in the notation of proposition 5.6.1:

N	1304	22940	69944
$\rho_\Gamma \omega_\Gamma, \Gamma \subset \Gamma_Q$	$9.72007 * 10^{-6}$	$3.27651 * 10^{-7}$	$1.12613 * 10^{-7}$
$\rho_{\partial K} \omega_{\partial K}$	0.444177	0.0160497	0.00551704

For $\kappa = 0.8$, these values are in the notation of proposition 5.6.1:

N	1208	11960	80492
$\rho_\Gamma \omega_\Gamma, \Gamma \subset \Gamma_Q$	$8.31925 * 10^{-6}$	$6.53136 * 10^{-7}$	$1.62319 * 10^{-7}$
$\rho_{\partial K} \omega_{\partial K}$	0.492626	0.0270697	0.00431046

The results show a reduction of the values from step to step. The terms on the whole domain are clearly dominant. This is also due to the fact that the domain Ω in this case is larger than in the previous examples. And for the mesh refinement, the value on each cell is important. The grids and the thereby obtained values for J show that the dual-weighted error estimator leads to a very good method for the placing of the vertices. Keep in mind that in this application J is not calculated by a boundary integral as in the previous cases. So the dominance of the domain integral has also a positive effect on the evaluation of J.

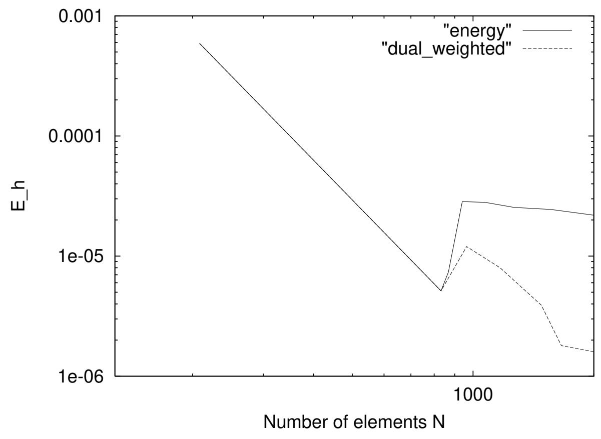


Figure 5.1: Drag optimization problem governed by Stokes equations: Comparison of efficiency of meshes generated by the energy-error estimator $\eta_E(u_h)$ (solid line) and the dual-weighted error estimator $\eta_w(u_h, \lambda_h, q_h)$ (dashed line) in log / log scale.

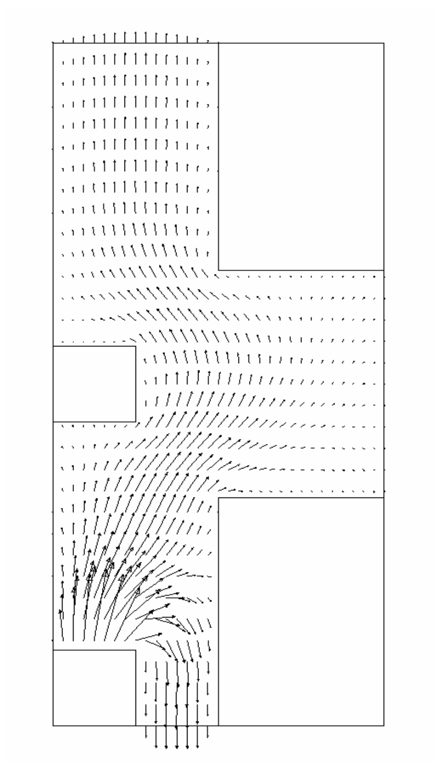


Figure 5.2: Drag optimization problem governed by Stokes equations: Obtained solutions for velocities w and u on 13000 cells.

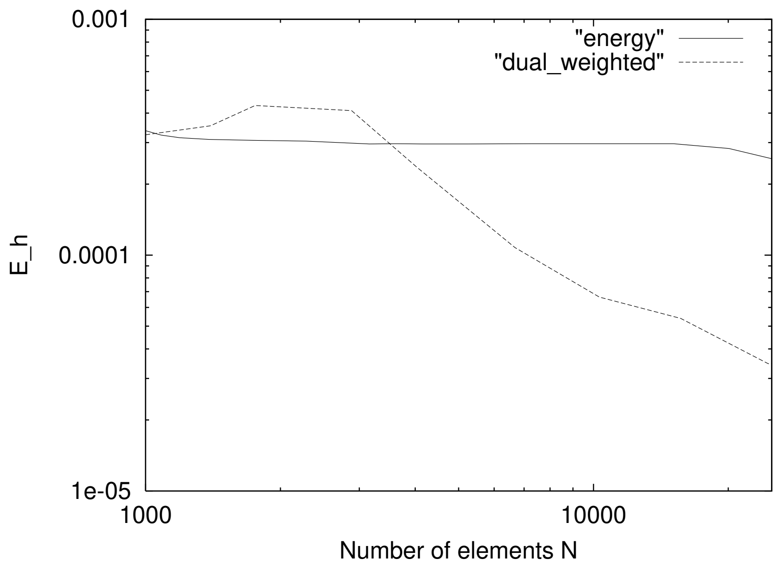


Figure 5.3: Drag optimization problem governed by Navier-Stokes equations with $\nu^{-1} = 15$: Comparison of efficiency of meshes generated by the energy-error estimator $\eta_E(u_h)$ (solid line) and the dual-weighted error estimator $\eta_w(u_h, \lambda_h, q_h)$ (dashed line) in log / log scale ($\alpha = 0.01$).

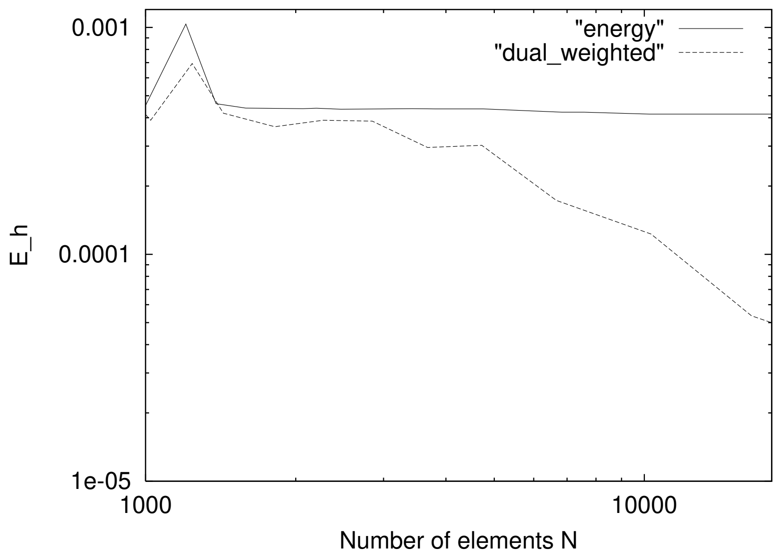


Figure 5.4: Drag optimization problem governed by Navier-Stokes equations with $\nu^{-1} = 20$: Comparison of efficiency of meshes generated by the energy-error estimator $\eta_E(u_h)$ (solid line) and the dual-weighted error estimator $\eta_w(u_h, \lambda_h, q_h)$ (dashed line) in log / log scale ($\alpha = 0.01$).

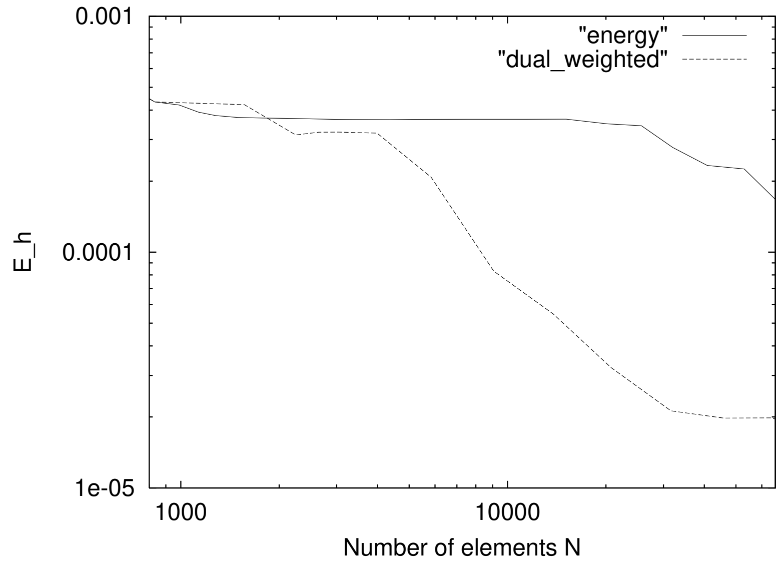


Figure 5.5: Drag optimization problem governed by Navier-Stokes equations with $\nu^{-1} = 20$: Comparison of efficiency of meshes generated by the energy-error estimator $\eta_E(u_h)$ (solid line) and the dual-weighted error estimator $\eta_w(u_h, \lambda_h, q_h)$ (dashed line) in log / log scale ($\alpha = 0.01$).

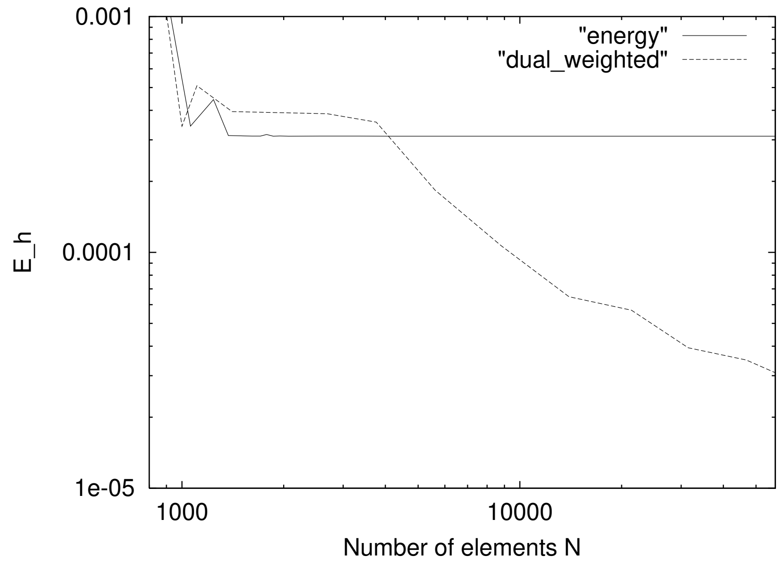


Figure 5.6: Drag optimization problem governed by Navier-Stokes equations with $\nu^{-1} = 15$: Comparison of efficiency of meshes generated by the energy-error estimator $\eta_E(u_h)$ (solid line) and the dual-weighted error estimator $\eta_w(u_h, \lambda_h, q_h)$ (dashed line) in log / log scale ($\alpha = 0.007$).

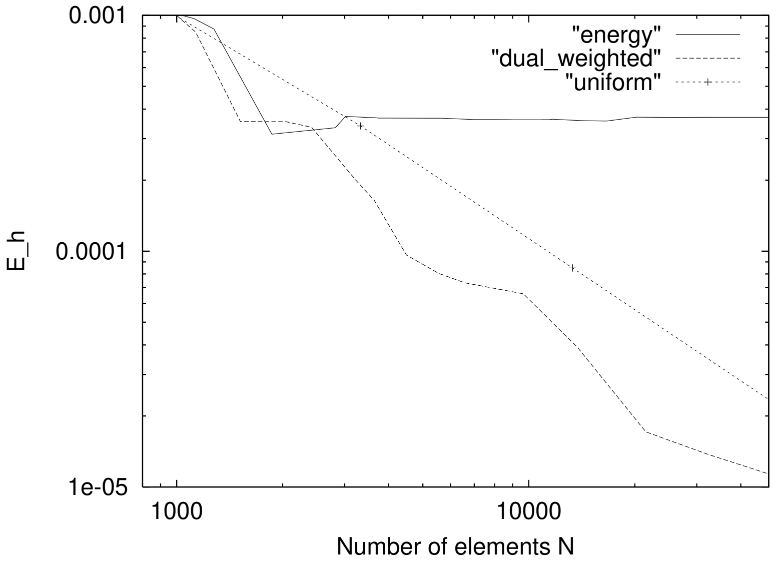


Figure 5.7: Drag optimization problem governed by Navier-Stokes equations with $\nu^{-1} = 70$: Comparison of efficiency of meshes generated by the energy-error estimator $\eta_E(u_h)$ (solid line), the dual-weighted error estimator $\eta_w(u_h, \lambda_h, q_h)$ (dashed line) and uniform refinement (crosses) in log / log scale ($\alpha = 0.007$).

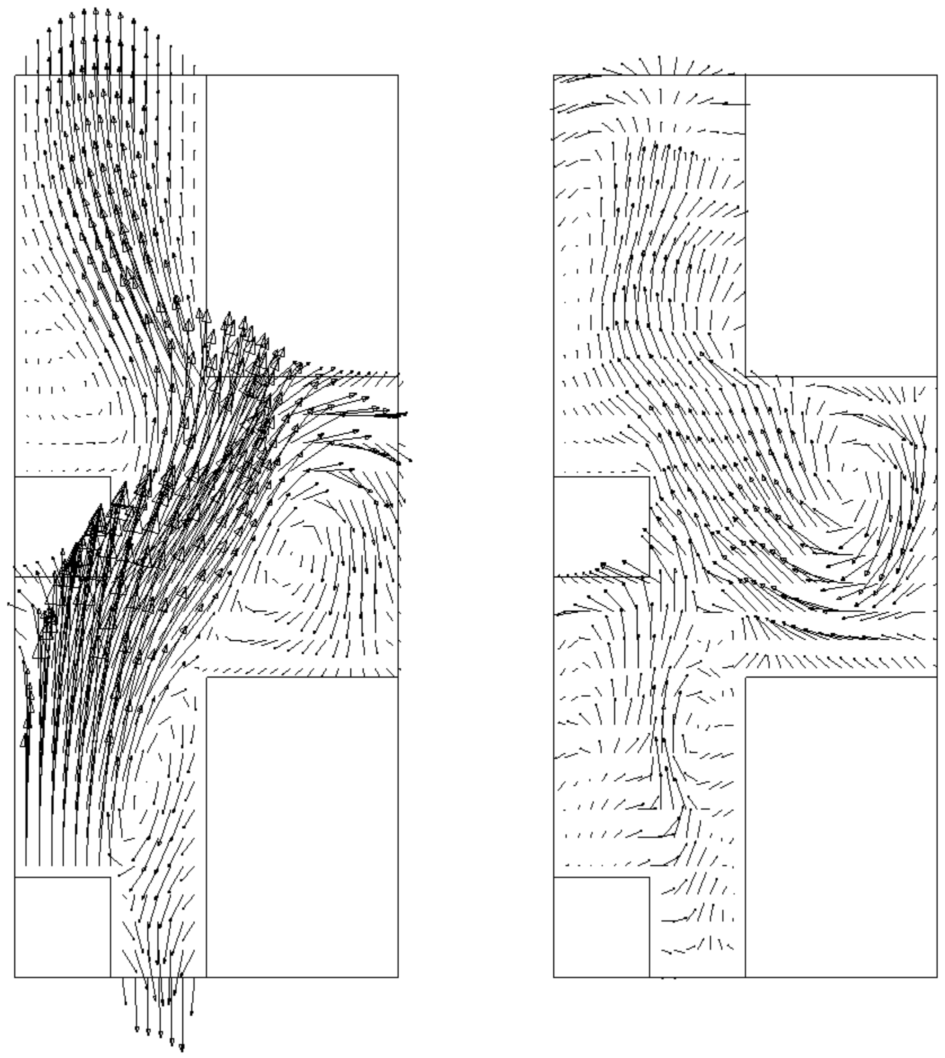


Figure 5.8: Drag optimization problem governed by Navier-Stokes equations: Numerical solutions for velocities u, w and corresponding Lagrangian multipliers λ_u, λ_w ($\nu^{-1} = 70, \alpha = 0.007$).

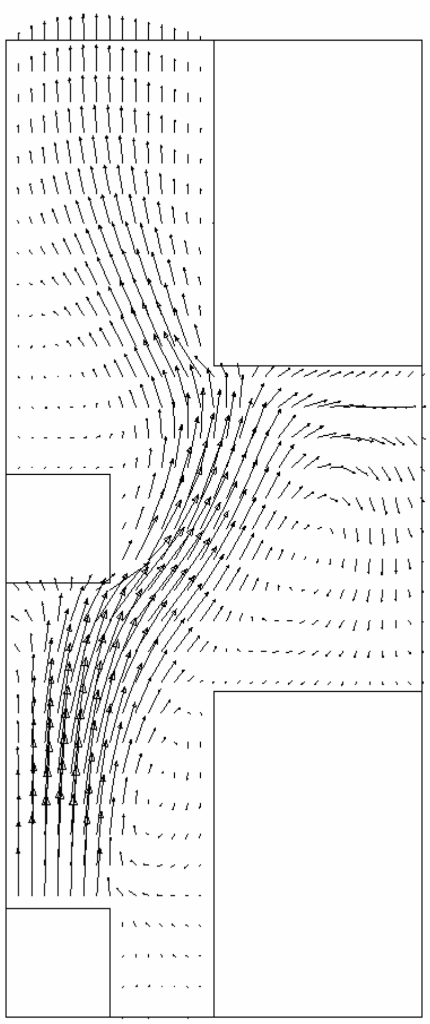


Figure 5.9: Drag optimization problem governed by Navier-Stokes equations: Numerical solutions for velocities u, w for fixed control $(q = 0)$ $(\nu^{-1} = 70, \alpha = 0.007)$.

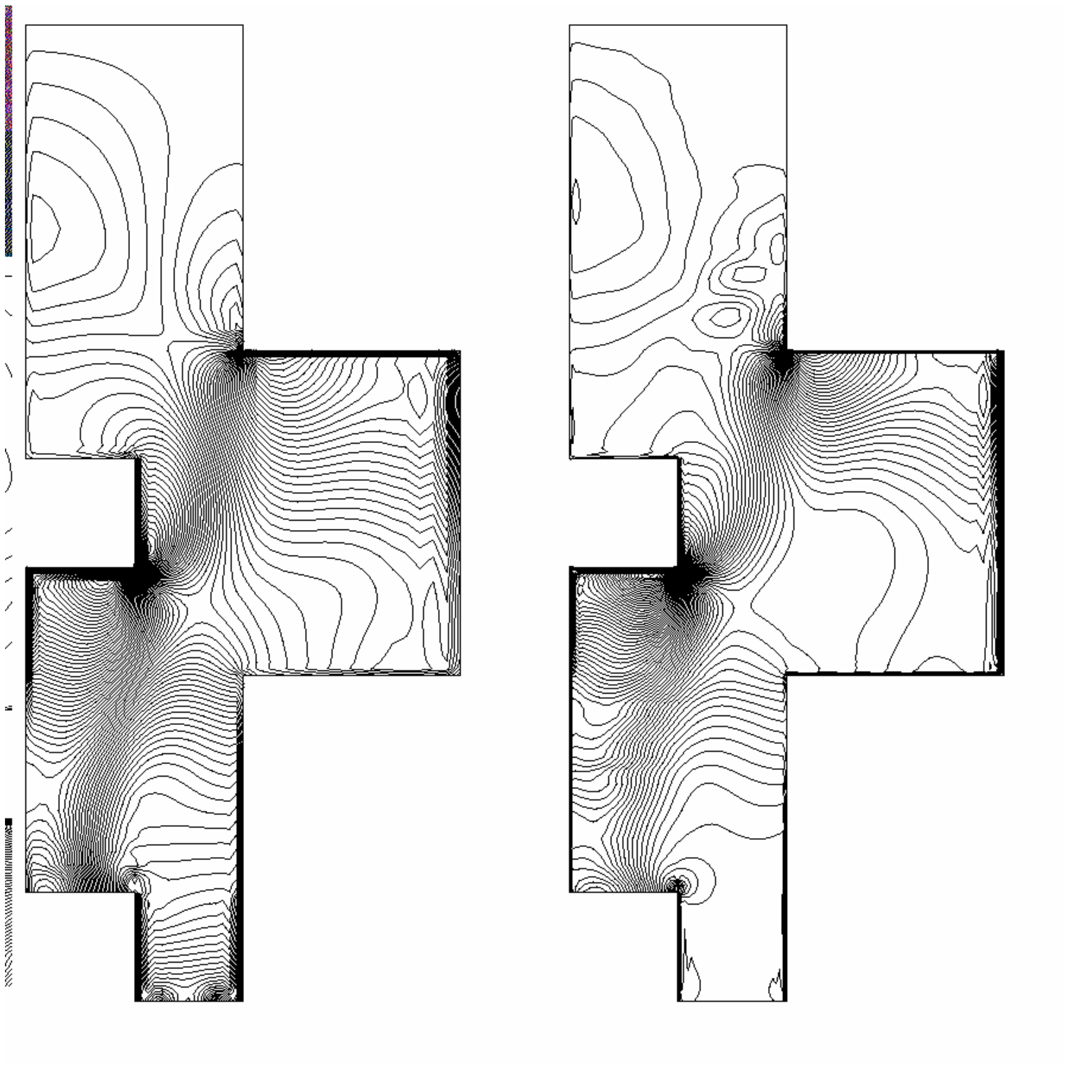


Figure 5.10: Drag optimization problem governed by Navier-Stokes equations: Comparison of pressure p for controlled flow and fixed control ($q = 0$) ($\nu^{-1} = 70, \alpha = 0.007$).

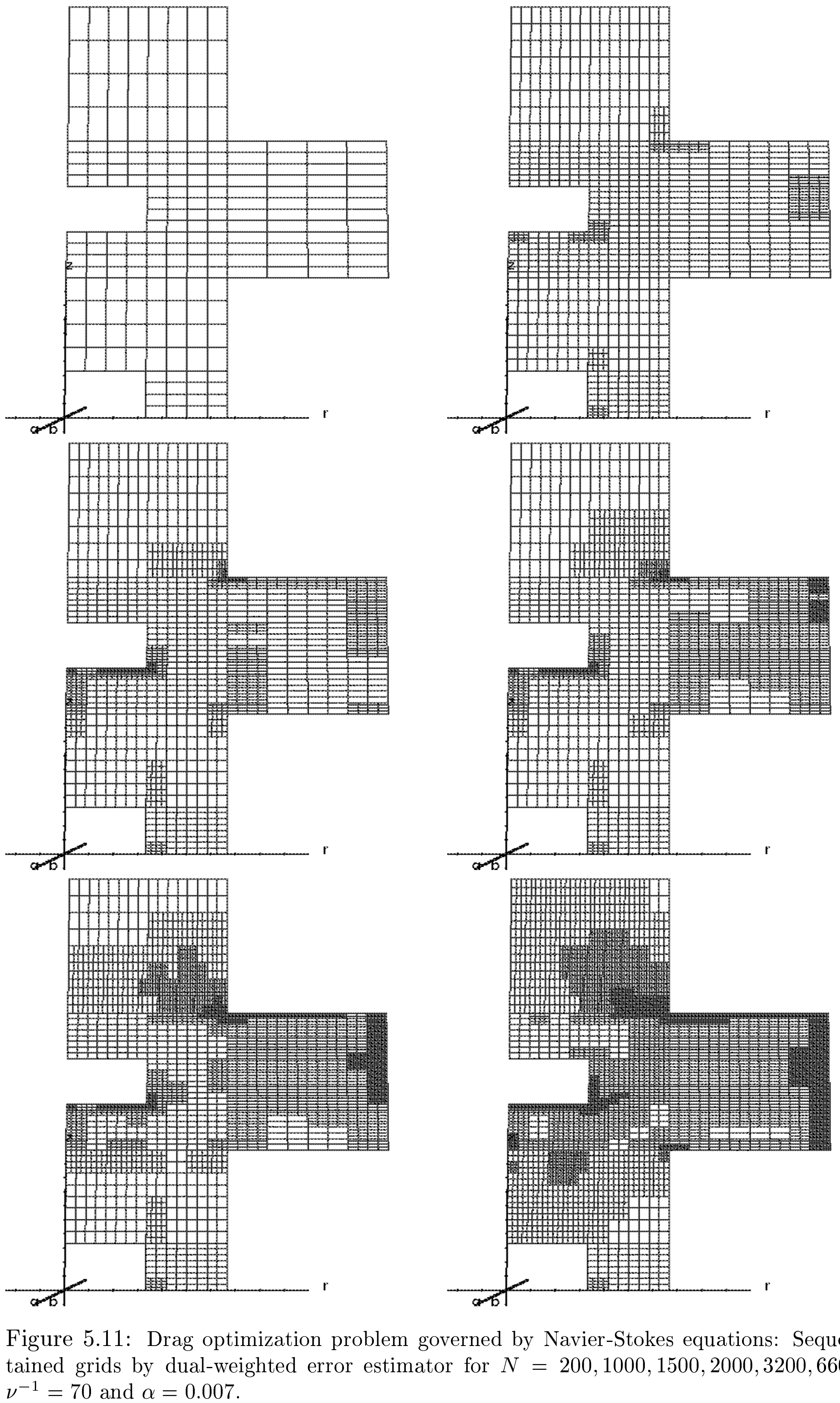


Figure 5.11: Drag optimization problem governed by Navier-Stokes equations: Sequence of obtained grids by dual-weighted error estimator for $N = 200, 1000, 1500, 2000, 3200, 6600$ cells for $\nu^{-1} = 70$ and $\alpha = 0.007$.

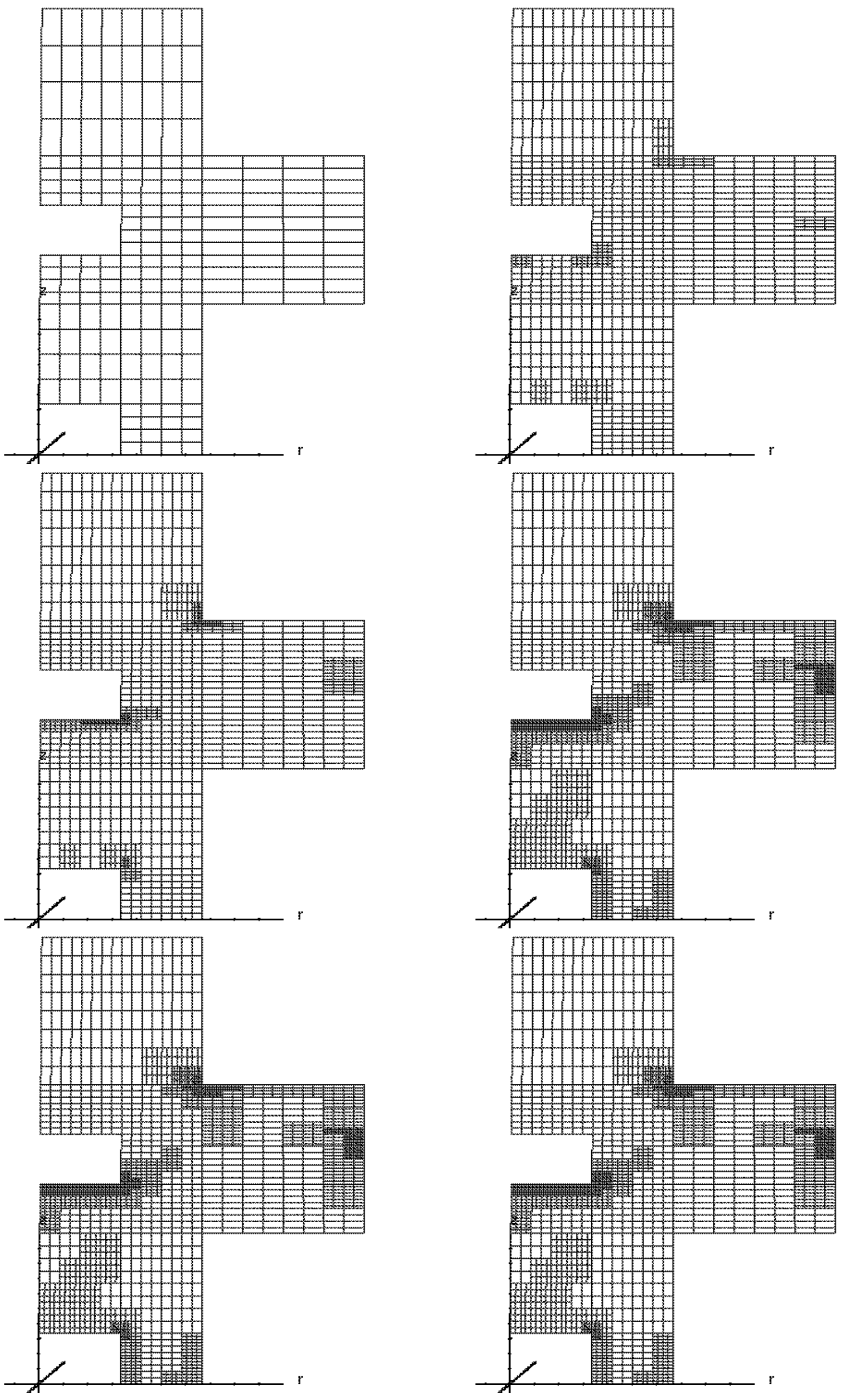


Figure 5.12: Drag optimization problem governed by Navier-Stokes equations: Sequence of obtained grids by energy-error estimator for $N = 200, 1000, 1300, 1800, 3200, 7000$ cells for $\nu^{-1} = 70$ and $\alpha = 0.007$.

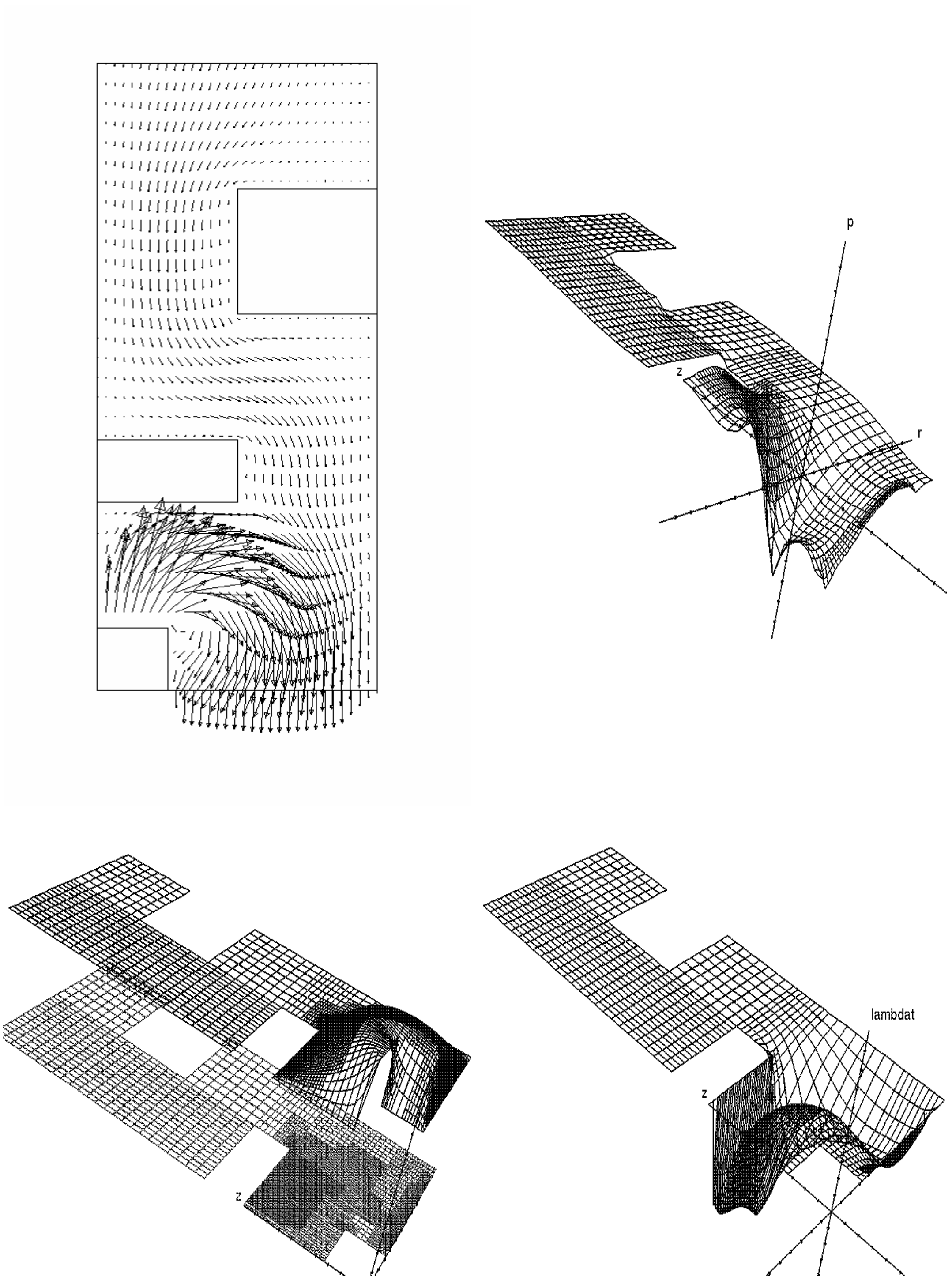


Figure 5.13: Temperature optimization problem governed by Navier-Stokes and Boussinesq equations: Obtained solutions for velocities, pressure p (above), temperature T and associated λ_T (below) with *dual-weighted error estimator* for $\nu^{-1} = 15$ and $\alpha = 0.01$, pure Newton method.

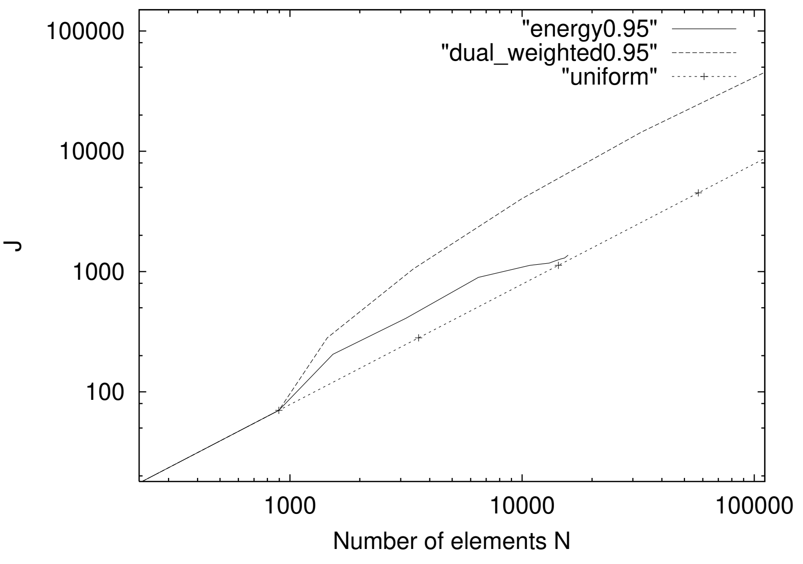


Figure 5.14: Temperature optimization problem governed by Navier-Stokes and Boussinesq equations: Comparison of efficiency of meshes generated by the energy error estimator $\eta_E(u_h)$ (solid line), the dual-weighted error estimator $\eta_w(u_h, \lambda_h, q_h)$ (dashed line) and uniform refinement (crosses) for $\kappa = 0.95$ in log / log scale ($\alpha = 0.01$).

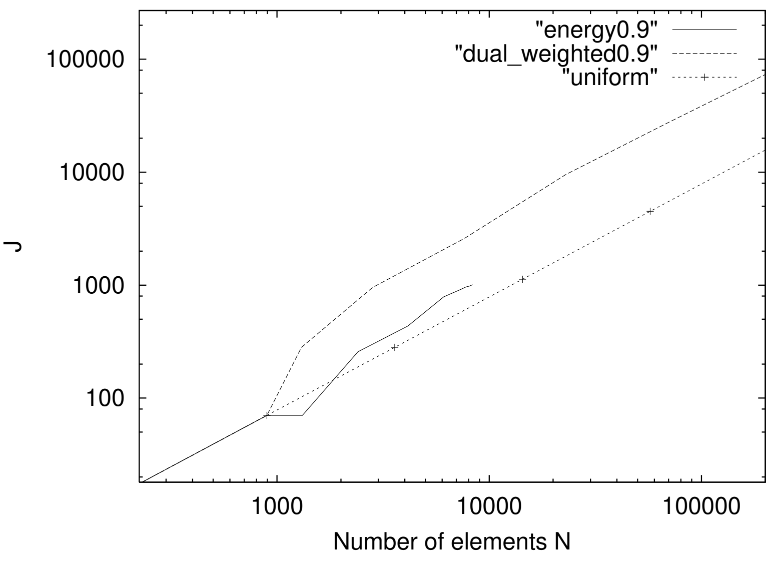


Figure 5.15: Temperature optimization problem governed by Navier-Stokes and Boussinesq equations: Comparison of efficiency of meshes generated by the energy error estimator $\eta_E(u_h)$ (solid line), the dual-weighted error estimator $\eta_w(u_h, \lambda_h, q_h)$ (dashed line) and uniform refinement (crosses) for $\kappa = 0.9$ in log / log scale ($\alpha = 0.01$).

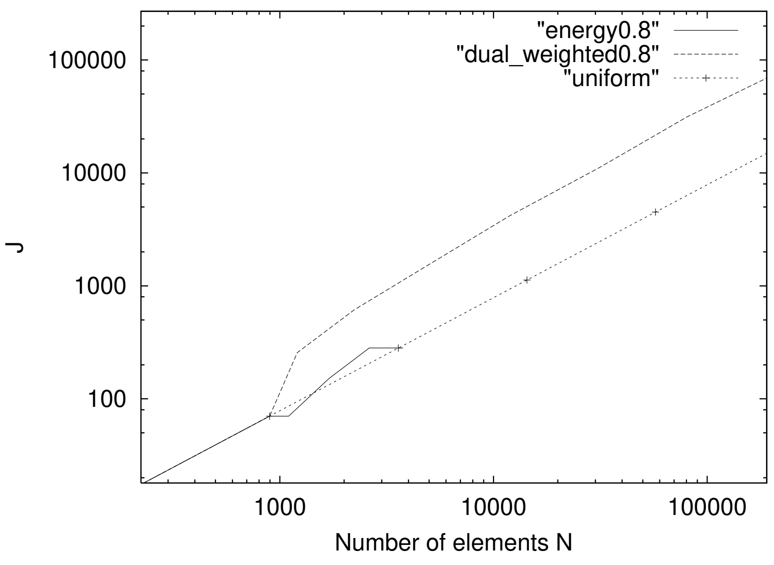


Figure 5.16: Temperature optimization problem governed by Navier-Stokes and Boussinesq equations: Comparison of efficiency of meshes generated by the energy error estimator $\eta_E(u_h)$ (solid line), the dual-weighted error estimator $\eta_w(u_h, \lambda_h, q_h)$ (dashed line) and uniform refinement (crosses) for $\kappa = 0.8$ in log / log scale ($\alpha = 0.01$).

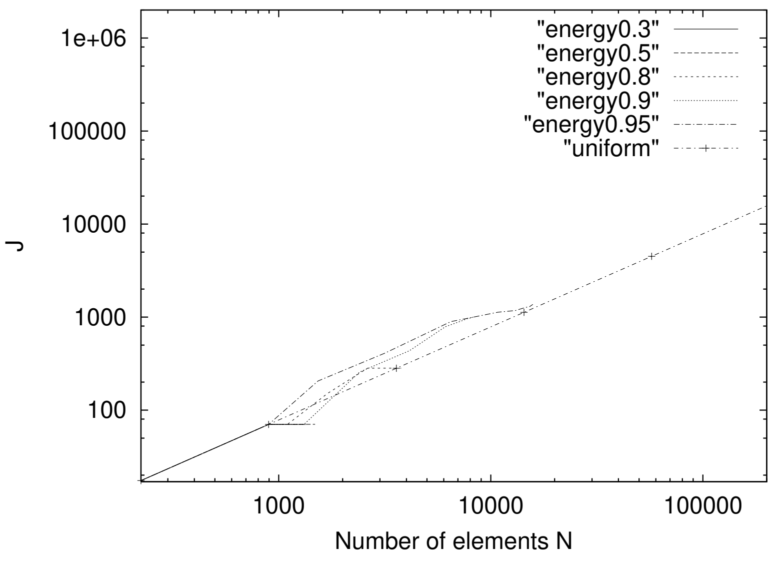


Figure 5.17: Temperature optimization problem governed by Navier-Stokes and Boussinesq equations: Comparison of efficiency of meshes generated by the energy error estimator $\eta_E(u_h)$ for $\kappa =$ 0.3, 0.5, 0.8, 0.9, 0.95 and uniform mesh refinement (crosses) in log / log scale ($\alpha = 0.01$).

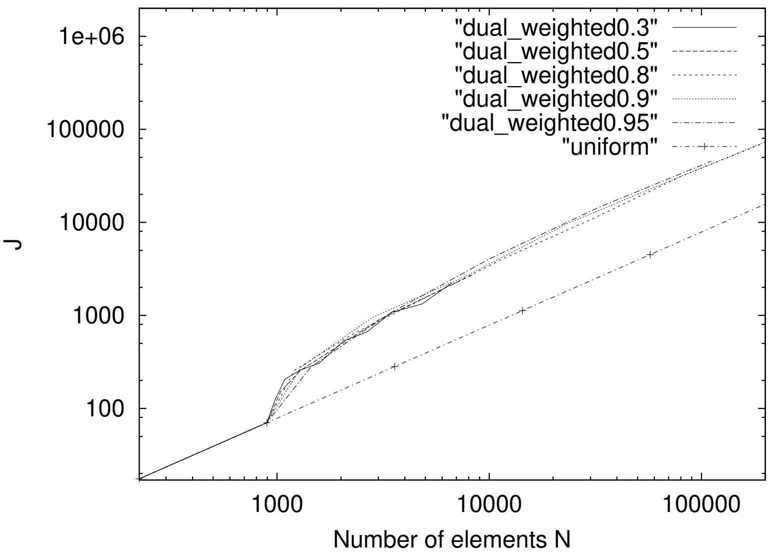


Figure 5.18: Temperature optimization problem governed by Navier-Stokes and Boussinesq equations: Comparison of efficiency of meshes generated by the dual-weighted error estimator $\eta_w(u_h, \lambda_h, q_h)$ for $\kappa = 0.3$, 0.5, 0.8, 0.9, 0.95 and uniform mesh refinement (crosses) in log / log scale ($\alpha = 0.01$).

Chapter 6

Numerical solution methods

In this chapter, some of the applied numerical techniques for solving the presented equations systems are described.

6.1 Solver

As described in section 1.1, the presented approach for solving an optimization problem governed by partial differential equations with adaptive finite element discretization results in algebraic systems. These equations are obtained by a Newton method approach (1.10) for all presented examples. The problem is to find a solver which is appropriate for the presented kind of problems. As described in section 6.2, this problem may involve several saddle points. Also the special structure from the finite element approach and the local mesh refinement has to be considered. There may be hanging nodes from the adaptive mesh refinement. Some methods have been tested like ordinary conjugate gradient, conjugate residual and several GMRES methods. The best results have been observed with GMRES. In some cases it was hard to get convergence at all.

The applied solver is a preconditioned GMRES method. The robustness of the solver is obtained from the GMRES while the acceleration of the convergence rate results from multi-grid.

This solver is constructed for linear problems. Hence, there must be a linearization of the derivation of the equation system. The linearization is done by the Newton-iteration on the continuous level. So already the continuous equation system is linearized. Therefore, the linearization problem is excluded from the derivation of the error estimator. Nevertheless, the linearization error can still emerge in the solution process. It is an independent error which may cause problems.

The presented globalization in Section 3.3 has several advantages. It enables a globalization of the Newton method. Furthermore, it provides a regularization as described in Section 3.3.

One of our developed codes is 'rhoptcon'. In this code exists the possibility to enable optimization without adaptivity if a certain exactness in the discrete system is obtained. By this, the optimization part can be disconnected from the adaptivity part. But for the codes 'bkr' and 'of' this is not necessary because the iteration is done to the limit of the

Newton iteration on each discrete level.

The accuracy level in the GMRES part can also be chosen with respect to the discretization level, especially depending on the number of vertices.

6.2 Preconditioner

For the applied GMRES method, a good preconditioner is necessary to accelerate the solution process. Multi-grid techniques have already been successfully applied with GMRES. The developed preconditioning technique uses information of the optimization problem. Calculation results show that this method is very successful. The convergence is getting much faster by this method.

Suppose, the optimization matrix has the following form

$$\begin{pmatrix} C_1 & C_2 & P^T \\ C_3 & C_4 & B^T \\ P & B & 0 \end{pmatrix}.$$

For the Ginzburg-Landau models in Chapter 4, we get the following coefficients (already applied on the variables as in Section 4.2):

$$\begin{aligned} C_1 &= (\delta u, \psi)_{obs} + (s''(u)\psi\delta u, \lambda)_\Omega \\ C_2 &= 0 \\ C_3 &= 0 \\ C_4 &= \alpha(\delta q, \chi)_{\Gamma_2} \\ P &= (\nabla\delta u, \nabla\phi)_\Omega + (s'(u)\delta u, \phi)_\Omega \\ B &= -(\delta q, \phi)_{\Gamma_2} \end{aligned}$$

The coefficients build the matrix of the scalar products of the base functions.

For controllability the term $u - u_d$ tends to zero. This means for the Hessian matrix, that the coefficient for $\frac{\partial^2 L}{\partial u^2}$ on the diagonal may lead to a numerical unstable preconditioner, because preconditioners often use the inverse of a matrix.

The diagonal entry for $\frac{\partial^2 L}{\partial q^2}$ can also lead to numerically unstable preconditioners. With regularization factor $\alpha = 0$, we get the original optimization problem. Choosing $\alpha > 0$ changes this problem. For $\alpha = 0$, the diagonal entry is zero for above derived problems. If α is small, we can also get some numerical problems with preconditioners.

The diagonal entry for $\frac{\partial^2 L}{\partial \lambda^2}$ is always zero, because λ is only linear in our equation system.

For these reasons, it seems better to choose an preconditioner which is based on the coefficient P. Here we get a well defined inverse if the simulation is well posed.

For the above optimization matrix, we get a precondition matrix like

$$\begin{pmatrix} 0 & 0 & P^{-1} \\ 0 & 0 & 0 \\ P^{-T} & 0 & 0 \end{pmatrix}. \tag{6.1}$$

Several tests confirm that if we take only the simulation and the dual solution for the precondition matrices P, we do not get a very good convergence. There is a progress to the simple choice of the whole optimization matrix because there are no zero-infinity entries on the diagonal, at least after using some stabilization techniques.

A better preconditioner is obtained by taking the first order differentiation of the simulation and the dual solution for the precondition matrices P. So we take the sensitivities of the simulation and the dual solution. Especially for an optimization problem, this choice can also be maintained for optimization focuses on these sensitivities. With this matrix, we perform a multi-grid step $u = P^{-1}\lambda$ and vice versa. This preconditioner leads to very good convergence results even for Navier-Stokes equations as simulation.

It should be mentioned that this precondition technique is also very successful for pure simulation by numerical results for Navier-Stokes equations.

For the Poisson equation as simulation, the following matrix P can be stated:

$$P = \left(\ (\nabla\chi, \nabla\psi) \ \right).$$

And for the Ginzburg-Landau equations in superconductivity as simulation, the matrix P results in:

$$P = \left(\ (\nabla\chi, \nabla\psi) + (s'(u) \cdot \chi, \psi) \ \right).$$

If the simulation are the Navier-Stokes equations, this matrix P has more entries which are also more complicated. This matrix can easily be derived from the given equations, because P is just the first order directional differentiation of the simulation or the part of the Hessian matrix of the Lagrange function which is indicated in (6.1).

For calculations with cylindrical polar coordinates, additional integrals motivated by the additional integrals arising from these coordinates can lead to an acceleration of the solution process. For example, the following integral is added:

$$\int_\Omega v_r \phi_r dr dz.$$

The boundary conditions are derived in section 6.4.

6.3 Symmetric discrete Hessian matrix

In the proof of Proposition 3.3.1 for the descent direction of a Newton method, the positive definiteness as well as the symmetry is used. This symmetry of the *discretized* Hessian matrix is not trivial. It depends on the chosen test and ansatz spaces. The presented approach in section 1.1 leads to an symmetric Hessian matrix of this type. The continuous primal and dual problems must be chosen as in sections 1.1 - 1.4 (i.e. some analytical subproblems are adjoint or transposed to each other) and the test and ansatz spaces must fit together as presented. It is much easier if the unknowns u, q, λ are calculated on the same grid. For a calculation on different grids, Petrov-Galerkin methods may be necessary. But in this case, an appropriate relation between the different grids has to be guaranteed. Otherwise, one looses symmetry and in some cases one will get some problems with the descent direction in the solution process of the optimization problem. The system matrix must be symmetrical. This is guaranteed due to a correct choice of the boundary conditions and the choice of the same bases in the test and ansatz spaces.

6.4 Calculation of the boundary conditions of the increments

For some solution methods, the boundary conditions of the increments are necessary. One example is the multi-grid which is applied in the preconditioner described in section 6.2.

For an optimal control problem, there are not only the increments of the simulation variables but also the increments of the Lagrangian multiplier and the control. As already mentioned in Section 1.4, there are relations between the boundary conditions of these variables. These relations will be used to calculate the boundary conditions. The boundary conditions follow automatically from the calculus developed in Section 1.4.

On the *control boundary*, the proper choice of the boundary condition of the increment of the variable δu would be $\delta u^n = u^n - u^*$ where u^n is the value of u in iteration n and u^* is the value of u in the optimum. The problem is that u^* is unknown, otherwise the optimal control problem would have already been solved. Therefore, the boundary condition for $\delta u^n = q^n - u^n$ in an iterative method is proposed where q^n is the value of the control q in iteration n. The motivation for this boundary conditions is: The boundary condition of u for (DBC) on the control boundary is $u = q$. Hence, the difference (which is δu) is $u - q$. In case of convergence we get $u = q$ and therefore $\delta u = 0$.

The boundary condition of the increment of the Lagrangian multiplier $\delta\lambda$ on the *control boundary* is derived as follows: As above, the boundary condition is in general $\delta\lambda^n = \lambda^n - \lambda^*$ where λ_n is the value of λ in iteration n and λ^* is the value of λ in the optimum. From the boundary condition $\lambda = 0$, it can be concluded that $\lambda^* = 0$. Hence, $\delta\lambda^n = \lambda^n$. This is the difference between the actual value and the value in the optimum.

On the *observation boundary*, there are natural Neumann boundary condition for u and λ. This applies to δu and $\delta\lambda$.

Remark for the DEAL library: It should be noted for the code that due to requirements of the DEAL library the Dirichlet and Neumann boundary conditions need not only be indicated in the function '`void CELL::set_boundary_line() const`' in the file '`line.cc`', but also in the file '`vertex.cc`'. The functions '`USERVertex::set_boundary`' and '`USERVertex::reset_boundary`' the boundary conditions of the increments must be set. This is caused by a special filter technique described in [58].

6.5 Calculation of Newton residuals and Newton increments

There are various ways to compute the values of the Newton residual and the Newton increments. The chosen evaluation is the value obtained by inserting the computed values of the discrete solution in the weak formulation of the Newton residual. These cell values are weighted by the Jacobi determinants on each cell. This determinant is the determinant of the transformation on the reference element. The weighting is therefore with geometric data. All is based on the weak formulations of the equations on the cells obtained from the adaptive finite element discretization.

The Newton increment is chosen in the same way.

In the whole optimization problem with adaptive finite element discretization, there are several residuals, which have to be evaluated to get good numerical results. Examples are the above Newton residuals or the residual from the solver GMRES. It is especially important that these residuals are weighted in an appropriate way. Otherwise the scaling of quantities in the algorithm is not balanced. For some residuals it is important to weight the values of the residual on each cell with the value of the Jacobi determinant of this cell. The general term 'residual' for several different terms appearing in such a complicated combination of methods may lead easily to misinterpretations and misunderstandings.

6.6 Calculation of differentials on the boundary

In this section, the chosen way of computing the values of the differentials on the boundary should be explained. This is especially important for (DBC), where this differential is a crucial part of the equation system of the control. An example is equation (1.21) in chapter 1. But this is also true for (DBC) in optimization for the Navier-Stokes equations.

The differentiation on the boundary in the direction of the domain is not well defined for variables defined only on the boundary like the control q. But for (DBC), the relation $u = q$ can be used. So u can be seen as a prolongation of q. Therefore one can transform certain properties of u to q, i.e. that u is well-defined.

The calculation of the value of the differential on the boundary in the direction of the domain for the control q uses the same idea. The value of the differential of u on the cell for which the control boundary is a part of the cell boundary is taken as the value of the differential of q. Important is that the weighting factor in the finite element calculation for the vertices on the control boundary is 0.25 due to the transformation of the value from the domain cell (calculated on a rectangular cell with 4 vertices).

This version of the calculation was the most stable. It enables an easy possibility to connect the values on the domain with the values on the boundary. Furthermore it is very cheap because this value already exists.

The presented method also connects the state equations with the (DBC) control q. By the translated differentiation information from u to q, also all conditions which are fulfilled by the state variable u are translated to the control q.

6.7 Implementation details

The code used in the test computations has been developed on the bases of the DEAL library (see Becker, Kanschat, and Suttmeier [6], [8], [43], [58]). DEAL is an object-orientated class library written in C++ which provides tools for the numerical solution of partial differential equations by adaptive finite element methods including multi-grid techniques. The developed codes 'rhoptcon', 'bkr' and 'of' are shortly organized as follows:

- Coarse grid construction: The structure of the code allows various possibilities for this basic coarse grid.

- Main loop of the program: Here, the adaptive mesh refinement already described in chapter 2 takes place. An adaptive mesh refinement is performed if a residual is small enough or after a maximal number of iterations. This loop is terminated if the value of the error indicator is smaller than a given tolerance TOL. For the codes 'rhoptcon' and 'bkr', a pure optimization part without adaptivity is possible afterwards.

- Selection of error indicators: For the adaptive mesh refinement part there exist several error indicators, which can easily be exchanged. This provides the possibility of effective comparisons between these indicators.

- Discretization level loop: The Newton method which is formally defined on the continuous level is evaluated on the discrete levels. In each Newton step a preconditioning by inversion of the state and dual state equations is applied (see section 6.2). This crucial step is done by GMRES which is accelerated by multi-grid.

For the globalization of the Newton method, there exist trust region methods or line searches as described in chapter 3. There are several types of line searches for the globalization of the Newton method, based on the Armijo-Goldstein principle or on modifications of it. Additionally, several merit functions for these line searches have been implemented. The methods have been developed for the special situation of optimization with partial differential equations and adaptive mesh refinement. We can also perform the pure Newton method without any globalization, which can lead to very good convergence rates for starting values close enough to the solution.

The code is designed to enable also calculations without regularization for the optimization problem, i.e., for $\alpha = 0$. This is done by calculating the incremental values for q by solving directly the equations for q in the full nonlinear system (1.7)-(1.9). Alternatively, for (NBC) in the applications Poisson equation and Ginzburg-Landau models, one may generate q from the trace of the Lagrangian multiplier, $\lambda|_{\Gamma_C} = \alpha q$. However, regularization was necessary in more complicated applications, e.g. optimization in flow problems.

Each code is splitted in several files grouping parts of the C++ code which belong to each other (respecting the object orientated class hierarchy). The main files are the following:

- The file '`delta.h`' contains the basic structure of the code.

- The C++ main program and some basic initializations are in '`main.cc`'.

- The file '`numeric.cc`' enables the management of some basic functions (invoking other functions with respect to the object orientated class hierarchy).

- The parts handling the boundaries are found in the file '`line.cc`'.

- Whereas the handling of the domain integrals are in '`quad.cc`'. Only rectangulars are used for the discretization of the finite elements in the domain.

- In the file '`cg_vector.h`' the information on the structure for the GMRES solver is contained.

- The file '`numcgv.cc`' gives some functions allowing the management of some functions connected to GMRES on the finite elements. They invoke several functions on the cells and lines.

- And the file '`dgmres.h`' contains this solver (originally contained in the DEAL library and slightly modified for optimization problems).

- The globalization of the Newton method described in the chapter 3 is principally done in the file '`global.cc`'.
- The file '`user.cc`' contains special functions for the problem formulation of the optimization problem. Examples are the (boundary) observations or regularization functions.
- In the file '`vertex.cc`' the handling of the vertices is organized. Here the interpolation functions or boundary values are defined.
- The functions for the grafic output are in '`grafic.cc`'.

Altogether, the for this thesis developed codes 'rhoptcon', 'bkr' and 'of' contain more than 7700, 7100 and 10000 lines, respectively (without the codes from the deal library).

The used compilers were the GNU compilers 'gcc' and 'g++'.

The presented graphical output is generated using CNOM, a graphic software developed by S. Krőmker at the SFB 359 in Heidelberg [46].

Appendix A

Nagopt - black box optimization code (for flow problems)

Black box optimization codes are commonly used for optimization. One standard library for black box optimization is the NAG library. Mark 14 and 15 in the Fortran version were used. Based on this library, the black box optimization code 'Nagopt' was developed by the author. It enables to use a given simulation code in C++, C or FORTRAN for optimization with almost no amendments.

The developed basic program 'Nagopt' is written in C++ and Fortran. By this code, the black box optimization algorithms of the NAG library are invoked. The applied NAG library routines are E04FCF, E04FDF, E04JBF, E04UCF and E04UPF. These NAG library functions are written in Fortran. The whole handling of data and functions is done by the basic program. The code is split in several files which are structured by their functionalities. The developed basic program 'Nagopt' contains more than 2000 lines.

The developed code will be applied to incompressible, laminar flow. The simulation code was provided by C. Waguet ([63]). The original 3D model was reduced to a 2D model by means of the rotational symmetry in the original 3D model. The obtained 2D flow tube has a reentrant corner.

The Navier-Stokes equations are formulated in cylindrical polar coordinates. This leads to the three components radial velocity u, axial velocity w and pressure p. The boundary conditions for the velocity are Dirichlet at the inflow and at the wall, on the other boundaries we have free boundary conditions (Neumann). Whereas for the pressure p, we require free boundary conditions on all boundaries.

The solution methods of the simulation are based on adaptive finite element methods using the weak formulation of the Navier-Stokes equations. For the nonlinear part, fix-point iteration is applied. The solver BiCGSTAB is accelerated by a multigrid preconditioning. The implementation is based on the DEAL code.

Calculations with Reynolds numbers Re in the range of $0.0001 \leq \mathrm{Re} \leq 100$ are possible. The parabolic inlet profile for the axial velocity is $\frac{1}{16}\,(x+4)(x-4)$.

Considering the optimization, there were two degrees of liberty in the problem:

1. Parameter estimation with respect to the *Reynolds number Re* or the *viscosity* ν.

2. Parameterization of the *parabolic inlet profile.*

The reference values for the parameter estimation are taken on the base grid.

Some information on the applied adaptivity: Normally, we have one or two adaptive refinements of the grid. For example, in level 0 we have 151 vertices and 120 cells. This is the starting grid. In level 1 we have 490 vertices and 429 cells. And in level 2 there can be 1506 vertices and 1377 cells. We use for each optimization step the same starting grid. The parameter estimation part was performed with black box algorithms from the NAG library.

The first considered optimization problem is **NAGOPT1**: Parameter estimation by choice of the viscosity or Reynolds number should be done. The reference values were obtained with $\nu^{-1} = 10$. There are 3 different sets of reference values, one with and two without adaptivity. All data points for the parameter estimation were already on the initial grid.

As starting values for ν^{-1}, 0.01 and 30 were used. The calculations were made both with and without adaptivity in the finite element part.

This leads to a least squares problem. Two black box algorithms for the NAG library were applied: The Gauss Newton method E04FCF and the Quasi-Newton method E04JBF.

In the first table, the accuracy in the parameter ν^{-1} was XTOL = 0.000001. The value of the objective function is given in the column 'final value'. The number of optimization iterations and simulation evaluations is denoted by '#iter' and '#sim', respectively.

algor	adaptiv	start v^{-1}	final ν^{-1}	final value	#iter	#sim
e04fcf - 1	no	0.01	9.99999	3.51e-16	3	12
e04fcf - 2	no	0.01	9.99999954	1.01e-16	3	12
e04fcf	0.01	0.01	10.00000	1.71e-14	2	10
e04fcf - 1	no	30	9.99999	3.51e-16	3	28
e04fcf - 2	no	30	9.99999956	1.62e-14	3	12
e04fcf	0.01	30	10.00000	1.71e-14	3	10
e04jbf - 1	no	0.01	9.99998807	3.53e-16	5	49
e04jbf - 2	no	0.01	9.99999800	1.63e-14	6	28
e04jbf	0.01	0.01	9.99999864	1.71e-14	7	30
e04jbf - 1	no	30	9.99998748	3.55e-16	4	56
e04jbf - 2	no	30	9.999999421	1.62e-14	6	48
e04jbf	0.01	30	9.99999835	1.71e-14	3	49

For larger XTOL no significant change was observed.

The Gauss Newton method E04FCF shows a much better behavior than the Quasi-Newton method E04JBF. The latter needs more iterations and more simulation evaluations. This leads to a slower solution process.

The second considered optimization problem is **NAGOPT2**: For this parameter estimation problem, we have the parameters viscosity/Reynolds number and the inlet factor for the inflow profile. The reference values were obtained with $\nu^{-1} = 10$ and inlet-factor 1. As above, there are 3 different sets of reference values, one with and two without adaptivity. All data point for the parameter estimation were already on the initial grid.

This leads to a least squares problem. Three black box algorithms for the NAG library were applied: The Gauss Newton method E04FCF, the Quasi-Newton method E04JBF and the SQP method E04UCF.

In the following table, the accuracy in the parameter ν^{-1} and in the inlet factor was XTOL = 0.000001.

algor	adaptiv	start ν^{-1}	start I	final value	#iter	#sim
e04fcf - 2	no	0.01	10.0	1.61e-14	8	30
e04fcf	0.01	0.01	10.0	1.68e-14	7	44
e04jbf - 2	no	0.01	10.0	1.61e-14	23	141
e04jbf	0.01	0.01	10.0	1.69e-14	20	124
e04ucf - 1	no	15.0	0.7	2.45e-15	18	24
e04ucf	0.01	15.0	0.7	1.84e-14	16	31
e04ucf - 1	no	0.9	4.0	5.35e-15	24	37
e04ucf - 2	no	0.01	10.0	1.62e-14	27	42

No convergence was stated for e04fcf-1, e04jbf-1 and e04ucf-1 for the starting values $\nu^{-1} = 0.01$ and I = 10.

Again, the Gauss Newton method E04FCF shows a much better behavior than the Quasi-Newton method E04JBF. But the SQP method E04UCF is even better.

The stated divergence in some cases shows that black box codes can not avoid difficulties which appear also for the developed methods in the main part of this thesis. But the latter allow a better treatment of these difficulties because more information of the simulation can be used in the optimization part due to the coupled system.

In general, the solution process is slower for the black box optimization codes. Every simulation needs a full solution of the forward system including its own adaptive mesh refinement. The differentiations are done by finite differences and are very expensive, especially for systems on fine grids. In the coupled system, this differentiation information is obtained directly from the system; it is therefore much cheaper.

Furthermore, no error estimates for the optimization problem can be developed with the black box version because the possible refinement in the simulation code can only be based on the simulation information. And the adaptivity must only be done one time for the developed method with coupled systems. For black box, adaptivity must be done in every simulation evaluation.

Appendix B

Equation system for optimization governed by the Navier-Stokes equations

In this appendix, the detailed equation system for optimization governed by the incompressible Navier-Stokes equations as application is derived. The presented terms are only those terms which result directly from the incompressible Navier-Stokes equations. Terms resulting from other parts of the optimization system like the cost functional, control terms or other boundary conditions must be added in correspondence with the special optimization problem.

The signs '+=' and '−=' mean adding or subtracting to the former value of the variable on the left hand side as used in computer science. N denotes the test functions.

The presented equations contain stabilization. For this reason the resulting differentials will lead to very large equation systems. The factor ξ enables to change the weighting of the transport term in the Navier-Stokes equations. Setting $\xi = 0$ eliminates the transport term. (In the code 'of' ξ is TRAP.)

For the *first order necessary conditions of the optimization problem*, the following equations result (for the Newton residual): From the continuity equation the following terms are obtained:

$$
\begin{aligned}
\lambda^{(0)} &= -\lambda_x^{(2)} N \\
\lambda^{(1)} &= -\lambda_y^{(2)} N
\end{aligned}
$$

From stabilization results:

$$
\begin{aligned}
\lambda^{(0)} &+= -\xi(u\lambda_x^{(2)} + u_x\lambda^{(2)} + w\lambda_y^{(2)})\tau N_x - \xi w_x \lambda^{(2)} \tau N_y \\
\lambda^{(1)} &+= -\xi u_y \lambda^{(2)} \tau N_x - \xi(u\lambda_x^{(2)} + w\lambda_y^{(2)} + w_y\lambda^{(2)})\tau N_y \\
\lambda^{(2)} &= -\lambda_x^{(2)} \tau N_x - \lambda_y^{(2)} \tau N_y
\end{aligned}
$$

From the first momentum equation (u) the following terms are obtained:

$$\begin{aligned}
\lambda^{(0)} &\ += \ -\xi(u\lambda_x^{(0)} + u_x\lambda^{(0)} + w\lambda_y^{(0)})N - \nu\lambda_x^{(0)}N_x - \nu * \lambda_y^{(0)}N_y \\
\lambda^{(1)} &\ -= \ \xi u_y\lambda^{(0)}N \\
\lambda^{(2)} &\ += \ \lambda^{(0)}N_x
\end{aligned}$$

From stabilization results:

$$\begin{aligned}
\lambda^{(0)} &\ -= \ \xi(u\lambda_x^{(0)} + u_x\lambda^{(0)} + w\lambda_y^{(0)})\tau(uN_x + wN_y) \\
\lambda^{(0)} &\ -= \ (\xi(uu_x\lambda^{(0)} + wu_y\lambda^{(0)}) + p_x\lambda^{(0)})\tau N_x \\
\lambda^{(1)} &\ -= \ \xi u_y\lambda^{(0)}\tau(uN_x + wN_y) \\
\lambda^{(1)} &\ -= \ (\xi(uu_x\lambda^{(0)} + wu_y\lambda^{(0)}) + p_x\lambda^{(0)})\tau N_y \\
\lambda^{(2)} &\ -= \ \lambda_x^{(0)}\tau(uN_x + wN_y)
\end{aligned}$$

From the second momentum equation (w) the following terms are obtained:

$$\begin{aligned}
\lambda^{(0)} &\ -= \ \xi w_x\lambda^{(1)}N \\
\lambda^{(1)} &\ -= \ \xi(u\lambda_x^{(1)} + w_y\lambda^{(1)} + w\lambda_y^{(1)})N \\
\lambda^{(1)} &\ -= \ \nu\lambda_x^{(1)}N_x + \nu\lambda_y^{(1)}N_y \\
\lambda^{(2)} &\ += \ \lambda^{(1)}N_y
\end{aligned}$$

From stabilization results:

$$\begin{aligned}
\lambda^{(0)} &\ -= \ \xi w_x\lambda^{(1)}\tau(uN_x + wN_y) \\
\lambda^{(0)} &\ -= \ (\xi(uw_x + ww_y)\lambda^{(1)} + p_y\lambda^{(1)})\tau N_x \\
\lambda^{(1)} &\ -= \ \xi(u\lambda_x^{(1)} + w_y\lambda^{(1)} + w\lambda_y^{(1)})\tau(uN_x + wN_y) \\
\lambda^{(1)} &\ -= \ (\xi(uw_x\lambda^{(1)} + ww_y\lambda^{(1)}) + p_y\lambda^{(1)})\tau N_y \\
\lambda^{(2)} &\ -= \ \lambda_y^{(1)}\tau(uN_x + wN_y)
\end{aligned}$$

The continuity equation is obtained by differentiation w.r.t. $\lambda^{(2)}$:

$$p \ = \ -(u_x + w_y)N$$

From stabilization results:

$$\begin{aligned}
p &\ -= \ p_x\tau N_x + p_y\tau N_y \\
p &\ -= \ \xi(uu_x + wu_y)\tau N_x + \xi(uw_x + ww_y)\tau N_y
\end{aligned}$$

The first momentum equation results from differentiation w.r.t. $\lambda^{(0)}$:

$$u \ = \ \xi(-uu_x - wu_y)N - \nu u_xN_x - \nu u_yN_y + pN_x$$

From stabilization results:

$$u \ -= \ (\xi uu_x + \xi wu_y + p_x)\tau(uN_x + wN_y)$$

The second momentum equation is obtained by differentiation w.r.t. $\lambda^{(1)}$:

$$w \;=\; \xi(-uw_x - ww_y)N - \nu w_x N_x - \nu w_y N_y + pN_y$$

From stabilization results:

$$w \;-=\; (\xi u w_x + \xi w w_y + p_y)\tau(uN_x + wN_y)$$

For the *Hessian matrix*, the following terms can be stated. The factor $\hat{\xi}$ denotes the weighting of the reaction term in the transport term of the Navier-Stokes equations. (In the code 'of' $\hat{\xi}$ is TRAPREACT.)

$\frac{\partial^{(2)} L(\vec{u},q,\vec{\lambda})}{\partial(\vec{u})\partial(\vec{u})}$: From continuity equation:

$$\begin{aligned}
\lambda^{(0)} &= \xi(\delta u_x \lambda^{(2)} + \delta u_x \lambda^{(2)})\tau N_x) + \xi \delta w_y \lambda^{(2)} \tau N_x + \xi \delta w_x \lambda^{(2)} \tau N_y \\
\lambda^{(1)} &= \xi \delta u_y \lambda^{(2)} \tau N_x + \xi \delta u_x \lambda^{(2)} \tau N_y) + \xi(\delta w_y \lambda^{(2)} + \delta w_y \lambda^{(2)})\tau N_y
\end{aligned}$$

From momentum equation 1 (u):

$$\begin{aligned}
\lambda^{(0)} &+= \xi(\delta u_x \lambda^{(0)} + \delta u_x \lambda^{(0)})N + \xi(\delta u_x \lambda^{(0)} + \delta u_x \lambda^{(0)})\tau(uN_x + wN_y) \\
\lambda^{(0)} &+= \xi u \delta u_x \lambda^{(0)} \tau N_x + \xi\hat{\xi} u_x \delta u \lambda^{(0)} \tau N_x + \xi(w \delta u_y \lambda^{(0)})\tau N_x \\
\lambda^{(0)} &+= \xi\hat{\xi} \delta u u_x \tau N_x + \xi u \delta u_x \lambda^{(0)} \tau N_x + \xi(w \delta u_y \lambda^{(0)})\tau N_x \\
\lambda^{(0)} &+= \xi \delta w_y \lambda^{(0)} N + \xi \delta w_y \lambda^{(0)} \tau(uN_x + wN_y) + \xi u \lambda^{(0)} \delta w_x \tau N_y \\
\lambda^{(0)} &+= \xi\hat{\xi} u_x \lambda^{(0)} \delta w \tau N_y + \xi(w \delta w_y \lambda^{(0)})\tau \delta w_y N_y + \xi\hat{\xi} \delta w u_y \lambda^{(0)} \tau N_x \\
\lambda^{(0)} &+= \delta p_x \lambda^{(0)} \tau N_x \\
\lambda^{(1)} &+= \xi \delta u_y \lambda^{(0)} N + \xi \delta u_y \lambda^{(0)} \tau(uN_x + wN_y) + \xi\hat{\xi} u_y \lambda^{(0)} \tau \delta u N_x + \xi u \delta u_x \lambda^{(0)} \tau N_y \\
\lambda^{(1)} &+= \xi\hat{\xi} \delta u u_x \lambda^{(0)} \tau N_y + \xi(w \delta u_y \lambda^{(0)})\tau N_y \\
\lambda^{(1)} &+= \xi\hat{\xi} u_y \lambda^{(0)} \tau \delta w N_y + \xi\hat{\xi} \delta w u_y \lambda^{(0)} \tau N_y \\
\lambda^{(1)} &+= \delta p_x \lambda^{(0)} \tau N_y \\
\lambda^{(2)} &= \lambda^{(0)} \tau \delta u_x N_x \\
\lambda^{(2)} &+= \lambda^{(0)} \tau \delta w_x N_y
\end{aligned}$$

From momentum equation 2 (w):

$$\begin{aligned}
\lambda^{(0)} &+= \xi\hat{\xi}\delta u w_x\lambda^{(1)}\tau N_x + \xi\hat{\xi}w_x\lambda^{(1)}\tau\delta u N_x \\
\lambda^{(0)} &+= \xi\delta w_x\lambda^{(1)}N + \xi\delta w_x\lambda^{(1)}\tau(uN_x + wN_y) + \xi\hat{\xi}w_x\lambda^{(1)}\tau\delta w N_y \\
\lambda^{(0)} &+= \xi u\delta w_x\lambda^{(1)}\tau N_x + \xi\hat{\xi}\delta w w_y\lambda^{(1)}\tau N_x + \xi(w\delta w_y)\lambda^{(1)}\tau N_x \\
\lambda^{(0)} &+= \delta p_y\lambda^{(1)}\tau N_x \\
\lambda^{(1)} &+= \xi\delta u_x\lambda^{(1)}N + \xi\delta u_x\lambda^{(1)}\tau(uN_x + wN_y) + \xi u\delta u_x\lambda^{(1)}\tau N_x \\
\lambda^{(1)} &+= \xi\hat{\xi}w_y\lambda^{(1)}\delta u\tau N_x + \xi(w\lambda^{(1)}\delta u_y)\tau N_x + \xi\hat{\xi}\delta u w_x\lambda^{(1)}\tau N_y \\
\lambda^{(1)} &+= \xi(\delta w_y\lambda^{(1)} + \delta w_y\lambda^{(1)})N + \xi(\delta w_y\lambda^{(1)} + \delta w_y\lambda^{(1)})\tau(uN_x + wN_y) \\
\lambda^{(1)} &+= \xi u\lambda^{(1)}\delta w_x\tau N_y + \xi\hat{\xi}w_y\lambda^{(1)}\delta w\tau N_y + \xi(w\lambda^{(1)}\delta w_y)\tau N_y \\
\lambda^{(1)} &+= \xi u\delta w_x\lambda^{(1)}\tau N_y + \xi\hat{\xi}\delta w w_y\lambda^{(1)}\tau N_y + \xi(w\delta w_y\lambda^{(1)})\tau N_y \\
\lambda^{(1)} &+= \delta p_y\lambda^{(1)}\tau N_y \\
\lambda^{(2)} &+= \lambda^{(1)}\tau\delta u_y N_x + \lambda^{(1)}\tau\delta w_y N_y
\end{aligned}$$

$\frac{\partial^{(2)}L(\vec{u},q,\vec{\lambda})}{\partial(\vec{\lambda})\partial(\vec{u})}$: From continuity equation:

$$\begin{aligned}
\lambda^{(0)} &+= (\delta\lambda^{(2)}_x)N + \xi(u\delta\lambda^{(2)}_x + w\delta\lambda^{(2)}_y)\tau N_x \\
\lambda^{(0)} &+= \xi\hat{\xi}u_x\delta\lambda^{(2)}\tau N_x + \xi\hat{\xi}w_x\delta\lambda^{(2)}\tau N_y \\
\lambda^{(1)} &+= \delta\lambda^{(2)}_y N + \xi\hat{\xi}u_y\delta\lambda^{(2)}\tau N_x \\
\lambda^{(1)} &+= \xi(u\delta\lambda^{(2)}_x + w\delta\lambda^{(2)}_y)\tau N_y + \xi\hat{\xi}w_y\delta\lambda^{(2)}\tau N_y \\
\lambda^{(2)} &+= \delta\lambda^{(2)}_x\tau N_x + \delta\lambda^{(2)}_y\tau N_y
\end{aligned}$$

From momentum equation 1 (u):

$$\begin{aligned}
\lambda^{(0)} &+= \xi(u\delta\lambda^{(0)}_x + w\delta\lambda^{(0)}_y)N + \xi\hat{\xi}u_x\delta\lambda^{(0)}N + \nu\delta\lambda^{(0)}_x N_x + \nu\delta\lambda^{(0)}_y N_y \\
\lambda^{(0)} &+= \xi u\delta\lambda^{(0)}_x\tau(uN_x + wN_y) + \xi\hat{\xi}u_x\delta\lambda^{(0)}\tau(uN_x + wN_y) \\
\lambda^{(0)} &+= \xi(w\delta\lambda^{(0)}_y)\tau(uN_x + wN_y) + \xi\hat{\xi}(uu_x\delta\lambda^{(0)} + wu_y\delta\lambda^{(0)})\tau N_x \\
\lambda^{(0)} &+= p_x\delta\lambda^{(0)}\tau N_x \\
\lambda^{(1)} &+= \xi\hat{\xi}u_y\delta\lambda^{(0)}N + \xi\hat{\xi}u_y\delta\lambda^{(0)}\tau(uN_x + wN_y) \\
\lambda^{(1)} &+= \xi\hat{\xi}(uu_x\delta\lambda^{(0)} + wu_y\delta\lambda^{(0)})\tau N_y + (p_x\delta\lambda^{(0)})\tau N_y \\
\lambda^{(2)} &-= \delta\lambda^{(0)}N_x - \delta\lambda^{(0)}_x\tau(uN_x + wN_y)
\end{aligned}$$

From momentum equation 2 (w):

$$\begin{aligned}
\lambda^{(0)} &+= \xi\hat{\xi}w_x\delta\lambda^{(1)}N + \xi\hat{\xi}w_x\delta\lambda^{(1)}\tau(uN_x + wN_y) \\
\lambda^{(0)} &+= \xi\hat{\xi}(uw_x + ww_y)\delta\lambda^{(1)}\tau N_x + p_y\delta\lambda^{(1)}\tau N_x \\
\lambda^{(1)} &+= \xi u\delta\lambda^{(1)}_x N + \xi\hat{\xi}w_y\delta\lambda^{(1)}N + \xi w\delta\lambda^{(1)}_y N \\
\lambda^{(1)} &+= \nu\delta\lambda^{(1)}_x N_x + \nu\delta\lambda^{(1)}_y N_y + \xi u\delta\lambda^{(1)}_x\tau(uN_x + wN_y) \\
\lambda^{(1)} &+= \xi\hat{\xi}w_y\delta\lambda^{(1)}\tau(uN_x + wN_y) + \xi(w\delta\lambda^{(1)}_y)\tau(uN_x + wN_y) \\
\lambda^{(1)} &+= \xi\hat{\xi}(uw_x\delta\lambda^{(1)} + ww_y\delta\lambda^{(1)})\tau N_y + (p_y\delta\lambda^{(1)})\tau N_y \\
\lambda^{(2)} &-= \delta\lambda^{(1)}N_y - \delta\lambda^{(1)}_y\tau(uN_x + wN_y)
\end{aligned}$$

$\frac{\partial^{(2)} L(\vec{u},q,\vec{\lambda})}{\partial(\vec{u})\partial(\vec{\lambda})}$: Continuity equation:

$$\begin{aligned}
p &\;=\; \delta u_x N + \xi\hat{\xi}\delta u u_x \tau N_x + \xi(u\delta u_x + w\delta u_y)\tau N_x + \xi\hat{\xi}\delta u w_x \tau N_y \\
p &\;+=\; \delta w_y N + \xi\hat{\xi}\delta w u_y \tau N_x + \xi(u\delta w_x + w\delta w_y)\tau N_y + \xi\hat{\xi}\delta w w_y \tau N_y \\
p &\;+=\; \delta p_x \tau N_x + \delta p_y \tau N_y
\end{aligned}$$

Momentum equation 1 (u):

$$\begin{aligned}
u &\;=\; \xi\hat{\xi}\delta u u_x N + \xi(u\delta u_x + w\delta u_y)N + \nu\delta u_x N_x + \nu\delta u_y N_y \\
u &\;+=\; \xi\hat{\xi}\delta u u_x \tau(uN_x + wN_y) + \xi u\delta u_x \tau(uN_x + wN_y) \\
u &\;+=\; \xi(w\delta u_y)\tau(uN_x + wN_y) + \xi\hat{\xi}(uu_x + wu_y)\tau\delta u N_x + p_x\tau\delta u N_x \\
u &\;+=\; \xi\hat{\xi}\delta w u_y N + \xi\hat{\xi}\delta w u_y \tau(uN_x + wN_y) \\
u &\;+=\; \xi\hat{\xi}(uu_x + wu_y)\tau\delta w N_y + p_x\tau\delta w N_y \\
u &\;-=\; \delta p N_x - \delta p_x \tau(uN_x + wN_y)
\end{aligned}$$

Momentum equation 2 (w):

$$\begin{aligned}
w &\;=\; \xi\hat{\xi}\delta u w_x N + \xi\hat{\xi}\delta u w_x \tau(uN_x + wN_y) + \xi\hat{\xi}(uw_x + ww_y)\tau\delta u N_x \\
w &\;+=\; p_y\tau\delta u N_x + \xi(u\delta w_x + w\delta w_y)N + \xi\hat{\xi}\delta w w_y N + \nu\delta w_x N_x + \nu\delta w_y N_y \\
w &\;+=\; \xi(u\delta w_x)\tau(uN_x + wN_y) + \xi\hat{\xi}\delta w w_y \tau(uN_x + wN_y) \\
w &\;+=\; \xi w\delta w_y \tau(uN_x + wN_y) + \xi\hat{\xi}(uw_x + ww_y)\tau\delta w N_y \\
w &\;+=\; p_y\tau\delta w N_y - \delta p N_y + \delta p_y \tau(uN_x + wN_y)
\end{aligned}$$

Acknowledgements

I want to thank Professor R. Rannacher for supervision of my Ph.D. thesis and valuable discussions on error estimation, numerics of partial differential equations and optimization. Especially, I want to thank for his support on several publications ([9], [10], [11], and [44]) and lectures on international conferences like ENUMATH97 and ENUMATH99. For the optimization part especially on globalization methods, I want to thank Professor H.G. Bock and Dr. J. Schlöder for discussions and hints.

I want to express special thanks to Dr. R. Becker for valuable teamwork on error estimation, optimization and Navier-Stokes equations.

My colleague C. Waguet I want to thank for good teamwork in our SFB359-project on reactive flows. I have to thank Dr. V. Heuveline and T. Dunne for revision of this thesis. The work group of Professor Wolfrum at the Physical-Chemical Institute of the University of Heidelberg was the partner for the cooperation concerning chemistry (reactive flows). Dr. T. Dreier and Dr. H.-R. Volpp contributed their advice. Thanks also to all concerned system administrators for providing the computer environment for programming and scientific computations on several computers and operation systems.

Special thanks to the DEAL developers Dr. R. Becker, Dr. G. Kanschat and Dr. F.-T. Suttmeier for providing and support of the DEAL library designed for computing numerical solutions of partial differential equations. This library is a crucial bases for my codes.

Furthermore, I want to thank the OPTPDE group for interesting discussions in our regular meetings on optimization and partial differential equations. I think that the different contributions lead to valuable knowledge for all participants, especially the hard core of these meetings besides me: Dr. V. Schulz, Dr. W. Egartner, H. Buss, and W. Bangert.

Special thanks also to Dr. S. Krömker for developing the graphic software 'cnom' [46]. Many of the presented pictures are due to cnom.

This work has been supported by the German Research Foundation (DFG), SFB 359 Reactive Flow, Diffusion and Transport, University of Heidelberg, INF 294, 69120 Heidelberg, Germany.

Errata

The following errata are added for the electronic publication. These are not contained in the version submitted to the Naturwissenschaftlichen - Mathematischen Gesamtfakultät der Ruprecht - Karls - Universität Heidelberg in April/May 2000.

Page 13, line 5: delete „, in Section 1.1“
Page 19, line 7: replace „$H^1(\Gamma;\Omega)$“ by „$H^1(\Gamma;\Omega)'$“
Page 19, line 21: replace „derivation“ by „differentiation“
Page 26, line 33: replace „to Hessian“ by „the Hessian“
Page 30, line 6: replace „Section 2.2“ by „Section 2.5“
Page 31, line 35-36: replace „in the last preceding paragraph“ by „above“
Page 34, line 6: replace „$-\Delta u$“ by $-\Delta u + u$“
Page 35, line 1: delete “continuous“ and „discrete“
Page 42, line 29: add after „, $r_h^{(q)}$“: „are the residuals obtained from the given functional considered in (2.6) and (2.7), these “
Page 46, line 21: replace „In section“ by „In this section“
Page 47, line 19: replace „not “ by „to“
Page 49, line 37: repace „optimization“ by „error estimation“
Page 56, line 2: replace „the“ by „of“
Page 64, line 2: delete „normally“
Page 65, line 30: replace second „x_k“ by „$\tilde{x_k}$“
Page 66, line 3 and 6: replace „$\triangle x_k\ \nabla^2 L(x_k)^{-1}$ “ by „$\triangle x_k^T\ \nabla^2 L(x_k)^{-1}$ “
Page 90, line 18: replace „bigger“ by „more complex“
Page 92, line 6: delete „can“
Page 93, line 31: replace „to“ by „two“
Page 98, line 11: delete „in“
Page 99, last line: replace „ϕ“ by „ψ„
Page 100, line 2: in second term: replace „ϕ“ by „ψ„

Bibliography

[1] M. Ainsworth and J.T. Oden: *A posteriori error estimation in finite element analysis*, Comp. Meth. Appl. Mech. Eng., 142 (1997).

[2] H.W. Alt: *Lineare Funktionalanalysis*, Springer-Verlag, 1985.

[3] J.R. Appel and M.D. Gunzburger: *Approximate solutions via sensitivities*, in *Inverse Design and Optimisation Methods*, Von Karman Institute for Fluid Dynamics, Lecture Series 1997-05, April 21-25 (1997).

[4] J. Arnold, T. Bouché, T. Dreier, J. Wichmann, and J. Wolfrum: *CARS studies on the heterogenous relaxation of vibrationally excited hydrogen and deuterium*, Chemical Physical Letters, 203 (1993).

[5] G.K. Batchelor: *An Introduction to Fluid Dynamics*, Cambridge University Press, 1967.

[6] R. Becker: *An Adaptive Finite Element Method for Incompressible Navier–Stokes Equations Time–dependent Domains*, Ph.D. Thesis Universität Heidelberg, 1995.

[7] R. Becker, M. Braack, R. Rannacher, and C. Waguet: *Fast and reliable solution of the Navier-Stokes equations including chemistry*, Comput. Visual. Sci., 2 (1999), pp. 107-122.

[8] R. Becker, G. Kanschat, and F.-T. Suttmeier: *Differential Equation Analysis Library*, documentation available on www, http://gaia.iwr.uni-heidelberg.de/˜deal/deal-v1/, 1997.

[9] R. Becker and H. Kapp: *Optimization in PDE models with adaptive finite element discretization*, Preprint 1998-20 IWR und SFB 359, University of Heidelberg (1998); Proc. ENUMATH'97, Heidelberg, Sept.29 - Oct.3, 1997, World Scientific Publ., Singapore, 1998.

[10] R. Becker, H. Kapp, and R. Rannacher: *Adaptive finite elements for optimal control of partial differential equations: Basic concept*, Preprint 1998-55 IWR und SFB 359, University of Heidelberg (1998); to appear in SIAM J. Control and Optimization.

[11] R. Becker, H. Kapp, and R. Rannacher: *Adaptive finite elements for optimization problems*, Preprint 1999-36 IWR und SFB 359, University of Heidelberg (1999); Proc. 18th Biennial Conf. on Numerical Analysis, June 29 - July 2, 1999, Dundee, CRC Press LLC, to appear.

[12] R. Becker and R. Rannacher: *Weighted a posteriori error control in FE methods*, ENUMATH-95, Paris, Sept. 18-22, 1995, in Proc. ENUMATH-97, World Scientific Publ., Singapore, 1998.

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

[14] H.G. Bock: *Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen*, Ph.D. Thesis Universität Bonn, 1987.

[15] H.G. Bock, J.P. Schlöder, and V.H. Schulz: *Numerik grosser Differentiell-Algebraischer Gleichungen - Simulation und Optimierung*, in: H. Schuler, Prozesssimulation, VCH, 1995.

[16] D. Braess: *Finite Elements*, Cambridge University Press, 1997.

[17] S.C. Brenner and R.L. Scott: *The Mathematical Theory of Finite Element Methods*, Springer-Verlag, 1994.

[18] G.F. Carey and J.T. Oden: *Finite Elements, Comptational Aspects*, III, Prentice-Hall, 1984.

[19] R.M. Chamberlain, M.J.D. Powell, C. Lemarechal, and H.C. Pedersen: *The watchdog technique for forcing convergence in algorithms for constrained optimization*, North-Holland Publishing Company, Mathematical Programming Study, 16 (1982), pp. 1-17.

[20] J.E. Dennis and R.B. Schnabel: *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prectice-Hall, 1983.

[21] J. Dieudonné: *Foundations of Modern Analysis*, Academic Press, London, 1969; *Éléments d'Analyse*, I, Gauthier-Villars Éditeur, 1972.

[22] J. Dieudonné: *Éléments d'Analyse*, II, Gauthier-Villars Éditeur, 1974; *Grundzüge der modernen Analysis*, 2, Vieweg, Braunschweig, 1987.

[23] J. Dieudonné: *Éléments d'Analyse*, III, Gauthier-Villars Éditeur, 1974; *Grundzüge der modernen Analysis*, 3, Vieweg, Braunschweig, 1976.

[24] Q. Du, M.D. Gunzburger, and J.S. Peterson: *Analysis and approximation of the Ginzburg-Landau model of superconductivity*, SIAM Review, 34 (1992), pp. 54-81.

[25] A.S. El-Bakry, R.A. Tapia, Y. Zhang, and T. Tsuchiya: *On the Formulation and Theory of the Newton Interior-Point Method for Nonlinear Programming*, Technical Report TR92-40, Rice University (1992, revised 1995).

[26] H.W. Engl, M. Hanke, and A. Neubauer: *Regularization of Inverse Problems*, Kluwer, Dodrecht, 1996.

[27] K. Eriksson, D. Estep, P. Hanspo, and C. Johnson: *Introduction to adaptive methods for differential equations*, Acta Numerica 1995 (A. Iserles, ed.), Cambridge University Press (1995), pp. 105-158.

[28] L.C. Evans: *Partial differential equations*, American Mathematical Society, 1998.

[29] L.P. Franca, S.L. Frey: *Stabilized finite element methods: II. The incompressible Navier-Stokes equations*, Comp. Methods Appl. Mech. Eng., 99 (1992), pp. 209-233.

[30] O. Forster, *Analysis 3*, vieweg studium, 52, Braunschweig, 1984.

[31] P.E. Gill, W. Murray, and M.H. Wright: *Practical Optimization*, Academic Press, 1992.

[32] M. Griebel, T. Dornseifer, and T. Neunhoeffer: *Numerische Simulation in der Strömungsmechanik*, vieweg Lehrbuch Scientific Computing, 1995.

[33] M.D. Gunzburger: *Introduction into mathematical aspects of flow control and optimization*, in *Inverse Design and Optimisation Methods*, Von Karman Institute for Fluid Dynamics, Lecture Series 1997-05, April 21-25 (1997).

[34] M.D. Gunzburger and L.S. Hou: *Finite-dimensional approximation of a class of constrained nonlinear optimal control problems*, SIAM J. Control and Optimization, 34 (1996), pp. 1001-1043.

[35] S.P. Han: *A Globally convergent method for nonlinear programming*, Journal of Optimization Theory and Applications, 22 (1977).

[36] D. Henry: *Geometric theory of semilinear parabolic equations*, Springer-Verlag, 1981.

[37] J.G. Heywood, R. Rannacher, and S. Turek: *Artificial boundaries and flux and pressure conditions for the incompressible Navier-Stokes equations*, International Journal for Numerical Methods in Fluids, 22 (1996), pp. 325-352.

[38] M. Hoza and M.A. Stadtherr: *Second-order correction methods for chemical process optimization*, Computers chem. Engng., 16 (1992), pp. 901-915.

[39] M. Hoza and M.A. Stadtherr: *An improved watchdog line search for successive quadratic programming*, Computers chem. Engng., 17 (1993), pp. 943-947.

[40] K. Ito, and K. Kunisch: *Augmented Lagrangian-SQP methods in Hilbert spaces and application to control in the coefficients problems.* SIAM J. Optimization, 6 (1996), pp. 96-125.

[41] K. Ito, and K. Kunisch: *Augmented Lagrangian-SQP methods for nonlinear optimal control problems of tracking type*, SIAM J. Control and Optimization, 34 (1996), pp. 874-891.

[42] C. Johnson: *Numerical Solutions of Partial Differential Equations by the Finite Element Method*, Cambridge University Press, 1987.

[43] G. Kanschat: *Parallel and Adaptive Galerkin Methods for Radiative Transfer Problems*, Ph.D. Thesis Universität Heidelberg, 1996.

[44] H. Kapp: *An approach to adaptivity in optimization problems*, Proc. ENUMATH'99, Jyvaeskylae, July 26-30, 1999, World Scientific Publ., to appear.

[45] H. Kardestuncer: *Finite Element Handbook*, McGraw-Hill, 1987.

[46] S. Krömker: *The Graphics Program cnom2.0, User Manual*, Preprint 95 - 32 Interdisziplinäres Zentrum für wissenschaftliches Rechnen, Universität Heidelberg (1995).

[47] D.B. Leineweber: *Efficient Reduced SQP Methods for the Optimization of Chemical Processes Described by Large Sparse DAE Models*, Ph.D. Thesis Universität Heidelberg, 1998.

[48] J.L. Lions: *Optimal Control of Systems Governed by Partial Differential Equations*, Springer-Verlag, Berlin, 1971.

[49] D.G. Luenberger: *Linear and Nonlinear Programming*, Addison-Wesley, 1989.

[50] O. Pironneau: *Optimal Shape Design for Elliptic Systems*, Springer-Verlag, New York, 1984.

[51] R. Rannacher: *Error control in finite element computations*, in Error Control and Adaptivity in Scientific Computing, edited by H. Bulgak and C. Zenger, NATO Science Series, C, 536, Kluwer Academic Publishers, 1999.

[52] M. Schäfer and S. Turek: *Benchmark Computations of Laminar Flow Around a Cylinder*, Preprint 1996-03 IWR und SFB 359, University of Heidelberg (1996).

[53] A. Schäfer: *Numerische Verfahren für grosse nichtlineare beschränkte Optimierungsprobleme und ihr Einsatz bei Optimal-Steuerungsproblemen*, Diploma Thesis Universität Heidelberg, 1999.

[54] R. Schmachtel: *Finite-Elemente-Diskretisierungen für dreidimensionale elliptische Randwertaufgaben mit Rotationssymmetrien*, Diploma Thesis Universität Heidelberg, 1995.

[55] J. Segatz: *Simulation und Optimierung von laminaren Strömungsreaktoren*, Ph.D. Thesis Universität Heidelberg, 1995.

[56] J. Segatz, R. Rannacher, J. Wichmann, C. Orlemann, T. Dreier, and J. Wolfrum: *Detailed numerical simulations in flow reactors: A new approach in measuring absolute rate constants*, J. Phys. Chem., 100 (1996), pp. 9323-9333.

[57] P. Spellucci: *Numerische Verfahren der nichtlinearen Optimierung*, Birkhäuser Verlag, 1993.

[58] F.-T. Suttmeier: *Adaptive Finite Element Approximation of Problems in Elasto-Plasticity Theory*, Ph.D. Thesis Universität Heidelberg, 1996.

[59] R. Temam: *Navier-Stokes Equations*, North-Holland, Elsevier Science Publishers B.V., 1984.

[60] M. Tinkham: *Introduction to Superconductivity*, McGraw-Hill, New York, 1975.

[61] S. Turek: *Efficient Solvers for Incompressible Flow Problems*, Springer-Verlag, 1999.

[62] R. Verfürth: *A Review of A Posteriori Estimation and Adaptive Mesh-Refinement Techniques*, Advances in Numerical Mathematics, Wiley/Teubner, New York-Stuttgart, 1996.

[63] C. Waguet: *An Adaptive Finite Element Code for Simulation of Chemical Flow Reactors*, Ph.D. Thesis Universität Heidelberg, 2000.

[64] A.J. Ward-Smith: *Internal Fluid Flow*, Clarendon Press, Oxford, 1980.

[65] F.M. White: *Viscous Fluid Flow*, McGraw-Hill, 1991.