Space-time Newton-multigrid strategies for nonstationary distributed and boundary flow control problems

Michael Hinze [*] Michael Köster[†‡] Stefan Turek [§]

October 13, 2013

Abstract

This paper considers a Newton-type solver strategy for optimal flow control problems using space-time multigrid solution techniques. Based on the standard Newton approach for optimal control, a space-time multigrid preconditioner is derived and numerically analysed for distributed and boundary control.

1 Introduction

The optimal control of incompressible, nonstationary flow problems belongs to today's most challenging problems in the field of optimisation. By design, all variables in the discretised equations are fully coupled, which drives these problems very challenging on the numerical level.

There are different discretisation and solver approaches available to approach nonstationary flow control, see [5, 11] for an overview. In [2, 3], the state of the art of multigrid methods in PDE constrained optimisation is summarised. Buildung upon [4] and the ideas promosed there, the present paper presents results for a multigrid-based solution strategy for the distributed and L^2 boundary control of nonstationary incompressible flow problems. A special Newton-type solver in the control space is developed which utilises space-time multigrid techniques for the Newton systems to enhance the efficiency.

In section 2 the model problems considered in this paper are introduced. Section 3 presents with a description of the standard Newton method in the control space and draws a comparison to other known methods. Section 4 introduces appropriate discretisation strategies for the space-time problems. The multigrid-based solver for the linear subproblems in the Newton approach is described in Section 5. Section 6 presents numerical tests regarding efficiency, and finally in Section 7, we draw some conclusions.

[*]Department of Mathematics, University of Hamburg, Bundesstrasse 55, 20146 Hamburg, Germany, michael.hinze@uni-hamburg.de

[†]corresponding author

[‡]Institute of Applied Mathematics, Technische Universität Dortmund, Vogelpothsweg 87, D-44227 Dortmund, Germany, michael.koester@mathematik.tu-dortmund.de

[§]Institute of Applied Mathematics, Technische Universität Dortmund, Vogelpothsweg 87, D-44227 Dortmund, Germany, stefan.turek@mathematik.tu-dortmund.de

2 Model problems

Our paper investigates distributed control as well as L^2 Dirichlet boundary control. In the following, let $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$ denote an open, bounded domain with boundary $\Gamma = \partial\Omega$ and outer normal vector η, $T > 0$ a final time, $\mathcal{Q} := (0, T) \times \Omega$ the corresponding space-time domain, and $\Sigma := (0, T) \times \Gamma$. Let the boundary Γ at each time instant be decomposed into the three different, disjoint parts Γ_D, Γ_N, Γ_C, with $\overline{\Gamma} = \overline{\Gamma}_D \cup \overline{\Gamma}_N \cup \overline{\Gamma}_C$. Γ_D specifies the Dirichlet part of the boundary, Γ_N the Neumann part, and Γ_C the Dirichlet control part. Furthermore, let $\Sigma_N := (0, T) \times \Gamma_N$, $\Sigma_C := (0, T) \times \Gamma_C$ and $\Sigma_D := (0, T) \times \Gamma_D$.

2.1 Optimal distributed control of the Navier–Stokes equations

Let $\Gamma_C = \emptyset$. Take a function $z : \mathcal{Q} \to \mathbb{R}^d$, the so-called target function, a Dirichlet boundary condition $g : (0, T) \times \Gamma_D \to \mathbb{R}^d$ and an initial condition $y^0 : \Omega \to \mathbb{R}^d$. The aim is to find a control $u : \mathcal{Q} \to \mathbb{R}^d$, a velocity field $y : \mathcal{Q} \to \mathbb{R}^d$ and a pressure field $p : \mathcal{Q} \to \mathbb{R}$ which solve the following minimisation problem.

$$J(y, u) = \frac{1}{2}||y - z||^2_{L^2(Q)} + \frac{\alpha}{2}||u||^2_{L^2(Q)} \quad \to \quad \min, \tag{1}$$

where y, p and u are coupled through the nonstationary Navier–Stokes equations,

$$\begin{aligned} y_t - \nu\Delta y + y\nabla y + \nabla p &= u && \text{in } \mathcal{Q}, \\ -\operatorname{div} y &= 0 && \text{in } \mathcal{Q}, \\ y &= g && \text{on } \Sigma_D, \\ \nu\partial_\eta y - p\eta &= 0 && \text{on } \Sigma_N, \\ y(0) &= y^0 && \text{in } \Omega. \end{aligned}$$

Using the Lagrange multiplier technique, the following corresponding KKT system can be derived,

$$\begin{aligned} y_t - \nu\Delta y + y\nabla y + \nabla p &= u & -\lambda_t + \nu\Delta y - y\nabla\lambda + (\nabla y)^T\lambda + \nabla\xi &= y - z && \text{in } \mathcal{Q}, \\ -\operatorname{div} y &= 0 & -\operatorname{div}\lambda &= 0 && \text{in } \mathcal{Q}, \\ y &= g & \lambda &= 0 && \text{on } \Sigma_D, \\ \nu\partial_\eta y - p\eta &= 0 & \nu\partial_\eta\lambda - \xi\eta + (y\eta)\lambda &= 0 && \text{on } \Sigma_N, \\ y(0) &= y^0 & \lambda(T) &= 0 && \text{in } \Omega, \\ & & \alpha u + \lambda &= 0 && \text{in } \mathcal{Q}. \end{aligned} \tag{2}$$

Here, $\lambda : \mathcal{Q} \to \mathbb{R}^d$ a dual velocity and $\xi : \mathcal{Q} \to \mathbb{R}$ and a dual pressure. It follows from (2) that in tis setting, the control u lives in the same space as the dual velocity λ.

2.2 Optimal L^2 boundary control of the Navier–Stokes equations

In the case of L^2 boundary control our minimisation problem reads: Find $u : \Sigma_C \to \mathbb{R}^d$, $y : \mathcal{Q} \to \mathbb{R}^d$ and $p : \mathcal{Q} \to \mathbb{R}$ which solve the following minimisation problem.

$$J(y, u) = \frac{1}{2}||y - z||^2_{L^2(Q)} + \frac{\alpha}{2}||u||^2_{L^2(\Gamma_C)} \quad \to \quad \min, \tag{3}$$

where y, p and u are coupled through the nonstationary Navier–Stokes equations,

$$\begin{aligned} y_t - \nu\Delta y + y\nabla y + \nabla p &= 0 && \text{in } \mathcal{Q}, \\ -\operatorname{div} y &= 0 && \text{in } \mathcal{Q}, \\ y &= g && \text{on } \Sigma_D, \\ y &= u && \text{on } \Sigma_C, \\ \nu\partial_\eta y - p\eta &= 0 && \text{on } \Sigma_N, \\ y(0) &= y^0 && \text{in } \Omega. \end{aligned}$$

Using the Lagrange multiplier technique, the following corresponding KKT system can be derived,

$$\begin{aligned} y_t - \nu\Delta y + y\nabla y + \nabla p &= 0 & -\lambda_t - \nu\Delta y - y\nabla\lambda + (\nabla y)^T\lambda + \nabla\xi &= y - z && \text{in } \mathcal{Q}, \\ -\operatorname{div} y &= 0 & -\operatorname{div}\lambda &= 0 && \text{in } \mathcal{Q}, \\ y &= g & \lambda &= 0 && \text{on } \Sigma_D, \\ y &= u & \lambda &= 0 && \text{on } \Sigma_C, \\ \nu\partial_\eta y - p\eta &= 0 & \nu\partial_\eta\lambda - \xi\eta + (y\eta)\lambda &= 0 && \text{on } \Sigma_N, \\ y(0) &= y^0 & \lambda(T) &= 0 && \text{in } \Omega, \\ \alpha u - (\nu\partial_\eta\lambda - \xi\eta) &= 0 && && \text{on } \Sigma_C. \end{aligned} \tag{4}$$

The control u acts only on the boundary and thus has a much smaller dimension than in the distributed case. However, for u to be computed, the fully coupled system has to be solved.

2.3 Control constraints

As an extension to the above problems, bounds can be introduced for the control. Let a distributed control u be bounded by

$$u_{\min} \le u \le u_{\max} \tag{5}$$

for some bounds $u_{\min}, u_{\max} : \mathcal{Q} \to \mathbb{R}^d$ in the distributed case or $u_{\min}, u_{\max} : \Sigma_C \to \mathbb{R}^d$ in the boundary control case, $u_{\min} < u_{\max}$. Using the L^2 projection operator P defined as

$$P(u) := \begin{cases} u_{\min}, & \text{where } u < u_{\min}, \\ u_{\max}, & \text{where } u > u_{\max}, \\ u, & \text{elsewhere.} \end{cases}$$

If we require (5) in the optimal control problem, according to, e.g., [16, Lemma 1.11], the control equation (2) in the distributed case changes to

$$u - P(u - \sigma(\alpha u + \lambda)) = 0 \qquad \text{in } \mathcal{Q}, \tag{6}$$

where $\sigma > 0$ can be chosen arbitrarily. The special choice $\sigma := \frac{1}{\alpha}$ leads to

$$u - P(-\frac{1}{\alpha}\lambda) = 0 \qquad \text{in } \mathcal{Q} \tag{7}$$

with the left-hand side being semismooth, see, e. g., [20]. In the boundary control case, one obtains

$$u - P(\frac{1}{\alpha}(\nu\partial_\eta\lambda - \xi\eta)) = 0 \qquad \text{on } \Sigma_C. \tag{8}$$

3 The integral equation method for nonlinear problems

The integral equation approach to solve our control problems is based on the control equations of the KKT system. At first, one defines the reduced cost functional

$$\hat{J}(u) := J(Su, u) \tag{9}$$

with $S : u \mapsto y$ being the solution operator that maps a control u to the solution y of the nonstationary Navier–Stokes equations. The first order optimality condition

$$(\hat{J}'(u), \bar{u} - u) \geq 0 \qquad \text{for all } \bar{u} \in U_{\text{ad}}$$

leads to the equations (2), (4), (7) and (8), respectively, which serve as starting point for this approach. From this, one derives a (semismooth) Newton method which we present exemplary for the control equations (2) and (7).

Unconstrained control In the unconstrained case the Newton iteration in the control space based on (2) reads

$$u_{n+1} := u_n - DF_1(u_n)^{-1}F_1(u_n), \tag{10}$$

i.e., expressed in two steps,

$$\begin{aligned} &\text{a.) solve} && DF_1(u_n)\bar{u}_n = d_n := -F_1(u_n) \\ &\text{b.) update} && u_{n+1} = u_n + \bar{u}_n \end{aligned}$$

with

$$F_1(u) := \hat{J}'(u) = \alpha u + \lambda \quad (\stackrel{!}{=} 0), \qquad\qquad DF_1(u)\bar{u} = \hat{J}''(u)\bar{u} = \alpha\bar{u} + \bar{\lambda}, \tag{11}$$

where (λ, ξ), (y, p) and $(\bar{\lambda}, \bar{\xi})$, $(\bar{y}, \bar{p})$ are the solutions of the following systems:

1.) Primal/dual equation

$$\begin{aligned} y_t - \nu\Delta y + y\nabla y - \nabla p &= u, & -\lambda_t - \nu\Delta y - y\nabla\lambda + (\nabla y)^T\lambda + \nabla\xi &= y - z && \text{in } \mathcal{Q}, \\ -\operatorname{div} y &= 0, & -\operatorname{div}\lambda &= 0 && \text{in } \mathcal{Q}, \\ y &= g, & \lambda &= 0 && \text{on } \Sigma_D, \\ \nu\partial_\eta y - p\eta &= 0, & \nu\partial_\eta\lambda - \xi\eta + (y\eta)\lambda &= 0 && \text{on } \Sigma_N, \\ y(0) &= y^0, & \lambda(T) &= 0 && \text{in } \Omega, \end{aligned}$$

2.) linearised primal equation

$$\begin{aligned}
\bar{y}_t - \nu\Delta\bar{y} + y\nabla\bar{y} + \bar{y}\nabla y + \nabla p &= \bar{u} && \text{in } \mathcal{Q},\\
-\operatorname{div}\bar{y} &= 0 && \text{in } \mathcal{Q},\\
\bar{y} &= 0 && \text{on } \Sigma_D,\\
\nu\partial_\eta\bar{y} - \bar{p}\eta &= 0 && \text{on } \Sigma_N,\\
\bar{y}(0) &= 0 && \text{in } \Omega,
\end{aligned}$$

3.) and linearised dual equation

$$\begin{aligned}
-\bar{\lambda}_t - \nu\Delta\bar{\lambda} - y\nabla\bar{\lambda} + (\nabla y)^T\bar{\lambda} + \nabla\bar{\xi} &= \bar{y} + \underbrace{\bar{y}\nabla\lambda - (\nabla\bar{y})^T\lambda}_{(12)} && \text{in } \mathcal{Q},\\
-\operatorname{div}\bar{\lambda} &= 0 && \text{in } \mathcal{Q},\\
\bar{\lambda} &= 0 && \text{on } \Sigma_D,\\
\nu\partial_\eta\bar{\lambda} - \xi\bar{\eta} + (y\eta)\bar{\lambda} &= -(\bar{y}\eta)\lambda && \text{on } \Sigma_N,\\
\bar{\lambda}(T) &= 0 && \text{in } \Omega.
\end{aligned}$$

Remarks a) The calculation of $F_1(u_n)$ involves the simulation of a nonlinear forward and a linear backward equation 1.) + 2.). The functions y_n and λ_n have to be stored.

b) The equation in step a.) is linear and can be solved with an iterative solver; in Section 5, we introduce a multigrid solver for this task. The iteration is based on the application of the operator $DF_1(\cdot)$, which involves the simulation of a linear forward and a linear backward equation 3.)/4.). Both problems can be solved at roughly the same costs. Thus, each Newton iteration amounts to one nonlinear forward simulation in 1.), one linear backward iteration in 2.) and a couple of linear forward and backward iterations for the linearised equations in 3.)/4.).

c) The term (12) was found to impose numerical difficulties in the first couple of Newton iterations. Numerical tests in this paper skip this term in the right-hand side assembly during the first one or two iterations, so the update is a kind of a mixture between a Picard and a Newton update.

Constrained control In the constrained case, we start from equation (7) and set

$$F_2(u) := u - P(-\frac{1}{\alpha}\lambda) \stackrel{!}{=} 0. \tag{13}$$

As this formula is not differentiable in the usual sense, we need a semismooth Newton method, whose corresponding iteration formula reads

$$\begin{aligned}
&\text{a.) Solve} && DF_2(u_n)\bar{u}_n = d_n := -F_2(u_n),\\
&\text{b.) Update} && u_{n+1} = u_n + \bar{u}_n.
\end{aligned}$$

Here, $DF_2(u_n)$ is an appropriate element of the subdifferential of F at u_n. $DF_2(\cdot)$ involves the generalised derivative $DP(\cdot)$ of the projection operator P. $DP(\cdot)$ set valued, cf. [16]. For

practical calculations, the following representative can be used and under certain conditions (see [19, 20]) is known to lead to superlinear convergence. For v, $\bar{v} : \mathcal{Q} \to \mathbb{R}^d$ let

$$DP(v)\bar{v} := \begin{cases} \bar{v} & \text{if } u_{\min} \le v \le u_{\max}, \\ 0 & \text{otherwise.} \end{cases}$$

With this setting, we obtain

$$DF_2(u_n)\bar{u} \;=\; \bar{u} - DP\big(-\frac{1}{\alpha}\lambda_n\big)\,\big(-\frac{1}{\alpha}\bar{\lambda}\big)$$

with $(\lambda_n, \xi_n) \equiv (\lambda, \xi)$, $(y_n, p_n) \equiv (y, p)$, and $(\bar{\lambda}, \bar{\xi})$, $(\bar{y}, \bar{p})$ the solutions of the (linearised) primal/dual systems from the unconstrained case above.

Remarks $DF_1(\cdot)$ is a symmetric operator as it is the Hessian of the reduced cost functional, see (11). With the above choice of DP, this also holds for $DF_2(\cdot)$. Thus, symmetric linear solvers like the Conjugate Gradient (CG) method can be used as solvers for the Newton system.

4 Discretisation

The discretisation of the optimal control problem is chosen in such a way that the optimise-then-discretise approach commutes with the discretise-then-optimise approach. We demonstrate this idea in a formal way based on (1). We use the following notations,

$$\begin{aligned} A(y) \quad &:= A(y,p) && := -\nu\Delta y + y\nabla y + \nabla p, \\ A'(y)\bar{y} \;&:= A'(y,p)(\bar{y},\bar{p}) && = -\nu\Delta\bar{y} + y\nabla\bar{y} + \bar{y}\nabla y + \nabla\bar{p}, \\ A'(y)^*\lambda &:= A'(y,p)^*(\lambda,\xi) && = -\nu\Delta\lambda - y\nabla\lambda + (\nabla y)^T\lambda + \nabla\xi. \end{aligned}$$

Choosing the rectangular rule for the discretisation of the cost functional and the implicit Euler scheme for the discretising of the primal equation leads to

$$J(\mathbf{y}^k, \mathbf{u}^k) = \frac{1}{2}k\sum_{i=1}^{N} ||y_i - z_i||_\Omega^2 + \frac{\alpha}{2}k\sum_{i=1}^{N} ||u_i||_\Omega^2,$$

where $\mathbf{y}^k = (y_0, ..., y_N)$, $\mathbf{u}^k := (u_0, ..., u_n)$ and

$$\begin{aligned} (y_i - y_{i-1}) + kA(y_i) &= ku_i && \text{in } \Omega, \\ -\operatorname{div} y_i &= 0 && \text{in } \Omega, \\ y_0 + kA(y_0) &= y^0 + kA(y^0) && \text{in } \Omega, \\ y_i &= g(t_i) && \text{on } \Gamma. \end{aligned}$$

Setting $\boldsymbol{\lambda}^k := (\lambda_0, ..., \lambda_n)$ and applying the formal Lagrange multiplier technique leads to

$$\begin{aligned} L(\mathbf{y}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k) \;:=\; J(\mathbf{y}^k, \mathbf{u}^k) &+ \sum_{i=1}^{N} \big(\lambda_i, ku_i - (y_i - y_{i-1} - kA(y_i))\big) && (14) \\ &+ \big(\lambda_0, (y^0 + kA(y^0)) - (y_0 - kA(y_0))\big), && (15) \end{aligned}$$

where boundary conditions are neglected for the moment. From $DL(\mathbf{y}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k) = 0$, one obtains the time-discretised system of equations,

$$\begin{aligned}
(y_i - y_{i-1}) + kA(y_i) &= ku_i, & (\lambda_i - \lambda_{i+1}) + kA'(y_i)^*\lambda_i &= k(y_i - z_i) && \text{in } \Omega,\\
-\operatorname{div} y_i &= 0, & -\operatorname{div} \lambda_i &= 0 && \text{in } \Omega,\\
y_0 + kA(y_0) &= y^0 + kA(y^0), & \lambda_N + kA'(y_N)^*\lambda_N &= k(y_N - z_N) && \text{in } \Omega,\\
y_i &= g(t_i), & \lambda_i &= 0 && \text{on } \Gamma,\\
& & \alpha u_i + \lambda_i &= 0 && \text{in } \Omega.
\end{aligned}$$

The discrete counterparts of the linearised primal/dual equations are derived by taking the Fréchet derivatives of the complete system,

$$\begin{aligned}
(\bar{y}_i - \bar{y}_{i-1}) + kA'(y_i)\bar{y}_i &= k\bar{u}_i, & (\bar{\lambda}_i - \bar{\lambda}_{i+1}) + kA'(y_i)^*\bar{\lambda}_i &= k(\bar{y}_i - \bar{z}_i) - kA'(\bar{y}_i)^*\lambda_i && \text{in } \Omega,\\
-\operatorname{div} y_i &= 0, & -\operatorname{div} \lambda_i &= 0 && \text{in } \Omega,\\
y_0 + kA(y_0) &= 0, & \bar{\lambda}_N + kA'(y_N)^*\lambda_N &= k(\bar{y}_N - \bar{z}_N) && \text{in } \Omega,\\
\bar{y}_i &= 0, & \bar{\lambda}_i &= 0 && \text{on } \Gamma,\\
& & \alpha\bar{u}_i + \bar{\lambda}_i &= 0 && \text{in } \Omega.
\end{aligned}$$

The time-discrete counterparts to $F(u)$ and $DF(u)\bar{u}$ can be derived as in Section 2.3.

The fully discretised system After applying the time discretisation, a space discretisation can be used to generate the fully discretised system. In the following, the fully discretised, vector valued variables are denoted by

$$\mathbf{y} := \mathbf{y}^{k,h} := (y_0^h, ..., y_N^h), \qquad \mathbf{u} := \mathbf{u}^{k,h} := (u_0^h, ..., u_N^h), \qquad \boldsymbol{\lambda} := \boldsymbol{\lambda}^{k,h} := (\lambda_0^h, ..., \lambda_N^h),$$

with y_k^h, u_k^h and λ_0^h vectors of degrees of freedom in the $\mathbb{R}^n$ in every timestep (for an appropriate n depending on the space). The index h indicates the discretisation in space and the index k the discretisation in time. In this work, we employ a discretisation with the Q_2 element for the velocity and the P_1^{disc} element for the pressure, see, e.g., [17].

The discrete (semismooth) Newton method The discrete counterpart of the (semismooth) Newton iteration used in this work reads

$$\text{1.) solve} \quad DF_{k,h}(\mathbf{u}_n)\mathbf{g} = \mathbf{d} := -F_{k,h}(\mathbf{u}_n), \tag{16a}$$

$$\text{2.) update} \quad \mathbf{u}_{n+1} := \mathbf{u}_n + \mathbf{g}, \tag{16b}$$

with

$$F_{k,h}(\mathbf{u}) := \alpha\mathbf{u} + \boldsymbol{\lambda}, \qquad DF_{k,h}(\mathbf{u})\bar{\mathbf{u}} = \alpha\bar{\mathbf{u}} + \bar{\boldsymbol{\lambda}} \tag{17}$$

in the unconstrained case and

$$F_{k,h}(\mathbf{u}) := \mathbf{u} - P_{k,h}(\mathbf{v}), \qquad \mathbf{v} := \mathbf{v}^{k,h} := -\frac{1}{\alpha}\boldsymbol{\lambda}, \tag{18}$$

$$DF_{k,h}(\mathbf{u})\bar{\mathbf{u}} := \bar{\mathbf{u}} - DP_{k,h}(\mathbf{v})\bar{\mathbf{v}}, \qquad \bar{\mathbf{v}} := \bar{\mathbf{v}}^{k,h} := -\frac{1}{\alpha}\bar{\boldsymbol{\lambda}} \tag{19}$$

in the constrained case. The discrete projection operator $P_{k,h}$ is realised by applying the projection to every degree of freedom. An approximation to its generalised derivative is

$$DP_{k,h}(\mathbf{v})\bar{\mathbf{v}} = (I_0\,\bar{v}_0^h,\ ...,\ I_N\,\bar{v}_N^h) \tag{20}$$

where the I_i are modified identity operators which set all those degrees of freedom in $\bar{v}_i^h$ to zero where the corresponding degree of freedom in v_i^h violates the bounds.

Remarks This realisation of the projection operator corresponds to a *first-optimise-then-discretise* approach which does not commute with *first-discretise-then-optimise* in general. Using Q_2 for the discretisation of the control, applying the projection only to the degrees of freedom will not guarantee the discrete control to be admissible. As a remedy, one can use a Taylor-Hood approach to discretise the control with piecewise linear continuous functions on a once refined mesh. Alternatively, one can realise the semismooth Newton method without discretisation of the control, see [12].

5 Multigrid for the control equation

Equation (16a) defines a linear system for the correction $\mathbf{g}$ of the control. The linear system is defined in space and time, each component of $\mathbf{g} = \mathbf{g}^{k,h} = (g_0^h, ..., g_N^h)$ defines one discrete function in space at a specified point in time. In this work, a multigrid approach according to Hackbusch [6–8] is applied to (16a) in order to solve this equation, see also [4]. This necessitates to begin with a couple of definitions.

5.1 The space-time mesh hierarchy

At first, we define a hierarchy of space-time meshes as follows:

- **Space hierarchy:** At first Ω_1, Ω_2,..., Ω_M denote a hierarchy of $M \in \mathbb{N}$ regularly refined meshes, i. e., new vertices are generated by connecting opposite midpoints.
- **Time hierarchy:** T_1,..., T_L describes a hierarchy of $L \in \mathbb{N}$ regularly refined meshes in time. The coarse mesh T_1 is assumed to contain $N_1 := N$ intervals of length $k_1 := \frac{1}{N}$, thus having $N + 1$ vertices. As a consequence, there are $N_l := 2^{l-1}N$ intervals in every T_m, each with timestep length $k_l := \frac{1}{N_l}$, and the time mesh has $2^{m-1}N + 1$ vertices.
- **Space-time hierarchy:** A space-time hierarchy can be created by different coarsening strategies, starting from the finest combination of space and time mesh. For simplicity, we assume $L = M$ and denote by $\mathcal{Q}_1$,...,$\mathcal{Q}_L$ a sequence of L nested space-time meshes. Typical choices for these hierarchies are, with $l = 1, ..., L$,
  - coarsening in space and time: $\mathcal{Q}_l := (T_l, \Omega_l)$,
  - semi-coarsening in time: $\mathcal{Q}_l := (T_l, \Omega_L)$,
  - semi-coarsening in space: $\mathcal{Q}_l := (T_L, \Omega_l)$.

5.2 Discretisation and problem hierarchy

Secondly, we have to define a discretisation on every level. In this work, the time discretisation is done with the Implicit Euler scheme, while the Q_2/P_1^{disc} finite element pair is used for the space discretisation. We use the following notations:

- V^m denotes for $m = 1, ..., L$ the space discretisation of the control space, realised by the degrees of freedom of the underlying finite element space.
- $W^{l,m}$ denotes for $l, m = 1, ..., L$ the space-time discretisation of the control space using V^m for the discretisation in space on the time scale T_l.
- W^l defines for $l = 1, ..., L$ the space-time discretisation of the control space corresponding to $\mathcal{Q}_l$. As a consequence, a coarsening strategy in space in time induces $W^l = W^{l,l}$, a semi-coarsening in time $W^l = W^{l,L}$ and a semi-coasening in space $W^l = W^{L,l}$.

Problem hierarchy a) The space-time discretisation induces a hierarchy of problems. For $l = 1, ..., L$, a hierarchy of discrete equations, derived from (10), reads

$$F_1^l(\mathbf{u}^l) := \alpha \mathbf{u}^l + \boldsymbol{\lambda}^l \stackrel{!}{=} 0 \qquad \text{in } W^l, \tag{21}$$

with $\mathbf{u}^l \in W^l$ the discrete solution to be determined, $\boldsymbol{\lambda}^l$ the corresponding dual solution to $\mathbf{u}$ and the operator $F_1^l = F_{k,h}$ the descrete counterpart to F_1 on W^l, with $k = k(l)$ and $h = h(l)$ identifying the mesh width on level l. Correspondingly, the discrete linearised system to be solved in every step of the Newton method on level l reads

$$DF_1^l(\mathbf{u}^l)\bar{\mathbf{u}}^l = -F_1^l(\mathbf{u})^l \qquad \text{in } W^l, \qquad DF_1^l(\mathbf{u}^l)\bar{\mathbf{u}}^l := \alpha\bar{\mathbf{u}}^l + \bar{\boldsymbol{\lambda}}^l. \tag{22}$$

with $\bar{\boldsymbol{\lambda}}^l$ the solution of the linearised discrete dual equation, corresponding to $\bar{\mathbf{u}}^l$. The solution of the problem is to be found on the finest mesh at level L.

b) In the constrained case, the discrete counterpart to (13) reads

$$F_2^l(\mathbf{u}^l) := \mathbf{u}^l - P^l(\mathbf{v}^l) \stackrel{!}{=} 0, \qquad \mathbf{v}^l := -\frac{1}{\alpha}\boldsymbol{\lambda}^l \tag{23}$$

in W^l, with $P^l = P_{k,h}$ define the descrete counterpart to P, and the discrete linearised system to be solved in every step of the semismooth Newton method on level l reads

$$DF_2^l(\mathbf{u}^l)\bar{\mathbf{u}}^l = -F_2^l(\mathbf{u})^l, \qquad DF_2^l(\mathbf{u}^l)\bar{\mathbf{u}}^l := \bar{\mathbf{u}}^l - DP^l(\mathbf{v}^l)\bar{\mathbf{v}}^l, \qquad \bar{\mathbf{v}}^l := -\frac{1}{\alpha}\bar{\boldsymbol{\lambda}}^l. \tag{24}$$

with DP^l the discrete counterpart to DP.

Primal/dual equations on lower levels Let $\mathbf{y}^L$ and $\boldsymbol{\lambda}^L$ define the solutions of the primal and dual equation on level L. For the operator DF^l to be applied on level $l < L$, corresponding primal/dual solutions $\mathbf{y}^l$ and $\boldsymbol{\lambda}^l$ are needed. They can be obtained with an L^2 projection of the finite element counterparts of $\mathbf{y}^L$ and $\boldsymbol{\lambda}^L$ in space and time to the lower level, realised approximately by a proper interpolation of the degrees of freedom.

5.3 Multigrid components

Multigrid needs a couple of components to be properly defined in order to be effective:

Prolongation Let the prolongation operator from level l to level $l+1$ be denoted by $I_l^{l+1} : W^l \to W^{l+1}$. Depending on the choice of the coarsening strategy, the operator has a time component and a spatial component. Each component u_i^h of a control vector $\mathbf{u}^l = (u_0^h, u_1^h, ..., u_{N_l}^h) \in W^l$ corresponds to a finite element function and thus, meaningful prolongation in space is the finite element prolongation. On the other hand, a prolongation in time is derived by a linear finite difference interpolation of the solutions in time, i. e.,

$$(u_0^h, u_1^h, ..., u_{N_l}^h) \mapsto \left(u_0^h, \frac{u_0^h + u_1^h}{2}, u_1^h, \frac{u_1^h + u_2^h}{2}, ..., u_{2N_l}^h \right).$$

Restriction Let a restriction operator from level l to level $l-1$ be denoted by $I_l^{l-1} : W^l \to W^{l-1}$, and let $d^l := (d_0^h, ..., d_{N_l}^h) \in W^l$ be a defect vector. Similar to the prolongation, the restriction has a time component and a spatial component. One possible choice for a restriction in time is a weighted mean in the sense of finite differences. However, for a discretisation with the implicit Euler, it is enough to apply a constant restriction in time, given by

$$(d_0^h, d_1^h, ..., d_{N_l}^h) \mapsto \left(d_0^h, \frac{d_1^h + d_2^h}{2}, d_2^h, \frac{d_3^h + d_4^h}{2}, ..., d_{N_l}^h \right).$$

This restriction is 'backward directed', thus, respects the direction of the propagation of information in time specified by the dual equation and will be shown to be effective in numerical tests.

For the restriction in space, one has to take into account that the discrete operator F^l maps $W^l \to W^l$, i. e., the operator works directly in the control space without any test functions, mass matrices or similar things involved. An appropriate choice is therefore the L^2 projection of the control space to a lower level, which is realised approximately by a simple interpolation of the degrees of freedom.

Coarse grid solver and smoother Typical choices for coarse grid solver and smoothers are iterative algorithms which only necessitate the application of the corresponding operator. For example, provided a damping parameter $0 < \omega \leq 1$, the Richardson iteration reads

$$\mathbf{g}_{\text{new}} := \mathbf{g} + \omega(\mathbf{d} - DF^l(\mathbf{u}_n)\mathbf{g}).$$

In a similar way, it is possible to apply a CG, BiCGStab or GMRES method. Applying such an algorithm on the coarse level until convergence is the usual choice for a coarse grid solver. Taking only a fixed number of iterations on any level except for the coarse level, one obtains a smoother for that level. In the following, we denote a smoother on level l applying NSM smoothing steps for a right-hand side $\mathbf{d}$ by the operator $\mathbf{g} \mapsto S_l(\mathbf{g}, \mathbf{d}, \text{NSM})$.

The operator to be applied in such algorithms reads DF^l and is realised by a forward-backward solving process: A forward iteration solves for $\bar{\mathbf{y}}$ and a backward iteration for $\bar{\boldsymbol{\lambda}}$. During the forward and the backward iteration, linear problems in space must be solved. In this work, we apply a multigrid solver in space for this task which provides low, level

independent convergence rates. Smoothing and coarse grid solving processes in space are realised with local pressure-Schur-Complement ('VANKA'-) like techniques which process all variables in space (velocity/pressure) simultaneously.

5.4 The multigrid algorithm

With the above components, Algorithm 1 describes a basic V-cycle multigrid in the control space. For a more general implementation (also concerning other cycles, etc.), the interested reader is referred to [1, 9, 21].

Algorithm 1 Space-time multigrid

Predefined constant: NSM $\in \mathbb{N}_0$: number of (post)smoothing steps

function SpaceTimeMultigrid($\bar{\mathbf{u}}$;$\mathbf{d}$;l)
  if ($l = 1$) **then**
    return $DF^l(\mathbf{u}^l)^{-1}\mathbf{d}$ ▷ coarse grid solver
  end if
  while (not converged) **do**
    $\mathbf{d}^{l-1} \leftarrow I_l^{l-1}(\mathbf{d} - DF^l(\mathbf{u}^l)\bar{\mathbf{u}}) \quad \in W^{l-1}$ ▷ restriction of the defect
    $\mathbf{g}^{l-1} \leftarrow$ SpaceTimeMultigrid($0; \mathbf{d}^{l-1}; l-1$) $\quad \in W^{l-1}$ ▷ coarse grid solution
    $\bar{\mathbf{u}} \leftarrow \bar{\mathbf{u}} + I_{l-1}^l(g^{l-1})$ ▷ coarse grid correction
    $\bar{\mathbf{u}} \leftarrow S_l(\bar{\mathbf{u}}, \mathbf{d}, \mathrm{NSM})$ ▷ postsmoothing
  end while
  return $\bar{\mathbf{u}}$ ▷ solution
end function

6 Numerical examples

As test examples, we consider the optimal control of a cavity flow and a fackward-facing step flow. Tests are carried out for single-grid solvers, multigrid solvers, for distributed control as well as for boundary control. However, the numerical experiments do not cover the case of control constraints. In the constrained case, the semismooth Newton method showed large convergence problems, even for a one-level solver. While the CG method worked fine after being restricted to the inactive set similar to [20], the semismooth Newton method oscillated between two states, not converging to the solution (using $\alpha = 0.01$, $[u_{\min}^{1,2}, u_{\max}^{1,2}] = [-0.5, 0.5]$). This problem could not be explained and solved during the time of the project, although other authors report this approach to work well, see [10, 20].

6.1 Distributed control for Driven–Cavity flow

Example We consider the optimal distributed control of the Navier–Stokes equations, see Section 2.1. The underlying domain is $\Omega = (0,1)^2$ with the four boundary parts Γ_1, Γ_2, Γ_3 and Γ_4 on the bottom, left, top and right. The problem is set up as a pure Dirichlet problem with $y(x,t) = (0,0)$ for $x \in \Gamma_1 \cup \Gamma_2 \cup \Gamma_4$ and $y(x,t) = (1,0)$ for $x \in \Gamma_3$. The coarse grid consists of only one square element. The time interval is defined as $[0,T]$ with $T = 1$, the viscosity parameter is set to $\nu = 1/400$. The initial flow y^0 is the stationary fully developed

Navier–Stokes flow at $\nu = 1/400$, while the target flow z is chosen as the fully developed, stationary Stokes flow, see Figure 1. The regularisation parameter for the control is set to $\alpha = 0.01$.

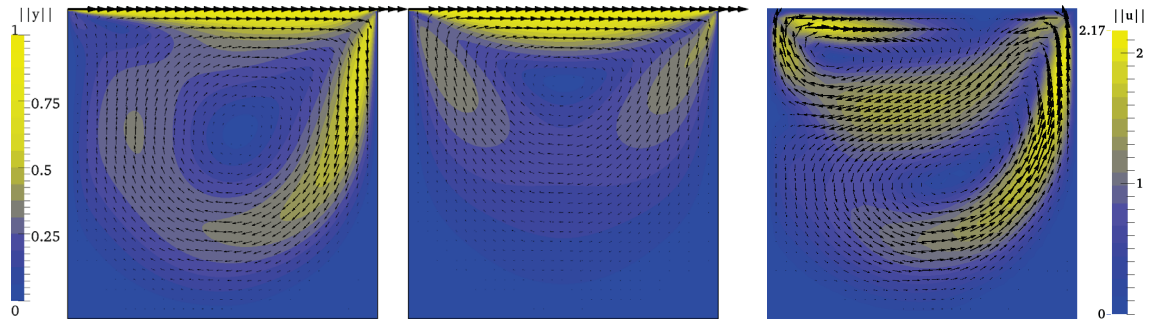

Figure 1: 'Driven–Cavity' example, velocity profile. Initial flow y^0 (left), target flow z (centre), optimal control u at $t = 0.0625$ (right).

The basic spatial coarse grid used in this test is a mesh containing one cell $[0, 1]^2$ three times refined, i. e., $h = 1/8$. The basic time mesh contains 20 time intervals. Both meshes are regularly refined to generate a hierarchy of meshes and a fine mesh used for the actual computation. Table 1 presents statistical data about the space and the time mesh for different refinement levels (with '#vertices' the number of vertices, '#edges' the number of edges, '#elements' the number of elements, $\#\text{dof}_{pd}$ the number of degrees of freedom in the primal and the dual space, $\#\text{dof}_c$ the number of degrees of freedom in the control space). As mentioned, the spatial discretisation is carried out with Q_2/P_1^{disc}.

Space-Level	#vertices	#edges	#elements	$\#\text{dof}_{pd}$	$\#\text{dof}_c$
4	81	144	64	770	578
5	289	544	256	2 946	2 178
6	1 089	2 112	1 024	11 522	8 450
7	4 225	8 320	4 096	45 570	33 282

Table 1: *Driven–Cavity:* Mesh statistics for distributed control, different refinement levels.

Solver configuration For the following tests we apply an inexact version of the described Newton algorithm above. The space-time Newton algorithm was configured to reduce the initial residual by six digits, while the space-time multigrid algorithm in every Newton step reduces its residual adaptively (at least gaining two digits) such that one obtains quadratic convergence; for a description of this strategy, see, e.g., [18]. The same stopping criterion was also used for the coarse grid solver. A V-cycle is used. For smoothing, four steps of a space-time CG method are applied.

Nonlinear and linear problems in space (calculated during the forward and backward loops) were solved until the L^2-norm of the residual drops below 10^{-14}; a spatial (Newton-)Multigrid solver with coarse grid solver on level four is applied in every timestep for this purpose. The local multigrid solver in space applies a local pressure Schur complement technique for smoothing and coarse grid solving, see also [13–15, 18]. This smoother is capable of processing

velocity and pressure variables simultaneously, which renders it ideal for saddle-point problems.

Solver efficiency test The following test applies a single grid and a multigrid solver strategy. On different refinement levels in space and time the Newton algorithm is applied, see Table 2. 'S.-Lv.' specifies the refinement level in space, '#int' the number of intervals in time, 'T_{opt}' documents the time which was necessary for the computation of the optimisation problem and 'T_{sim}' the time which was needed for the computation of the first forward simulation, i. e., for a simulation without any control applied. '#NL' and '$\sum$#LIN' depict the number of Newton steps and the sum of all steps of the linear solver, respectively.

Single grid CG preconditioner

S.-Lv.	#int	T_{opt}	T_{sim}	#NL	$\sum$#LIN	$\frac{T_{\text{opt}}}{T_{\text{sim}}}$	$\frac{\sum \#\text{LIN}}{\#\text{NL}}$
5	40	0:16:35	0:00:13	4	67	79.0	16.8
6	80	2:14:23	0:01:41	4	64	79.6	16.0
7	160	15:37:20	0:11:33	4	63	81.1	15.8

Multigrid preconditioner, pure space coarsening

S.-Lv.	#int	T_{opt}	T_{sim}	#NL	$\sum$#LIN	$\frac{T_{\text{opt}}}{T_{\text{sim}}}$	$\frac{\sum \#\text{LIN}}{\#\text{NL}}$
5	40	0:41:32	0:00:13	4	17	197.8	4.2
6	80	3:48:29	0:01:41	4	14	135.3	3.5
7	160	25:48:41	0:11:33	4	16	134.0	4.0

Multigrid preconditioner, pure time coarsening

S.-Lv.	#int	T_{opt}	T_{sim}	#NL	$\sum$#LIN	$\frac{T_{\text{opt}}}{T_{\text{sim}}}$	$\frac{\sum \#\text{LIN}}{\#\text{NL}}$
5	2	0:26:47	0:00:13	4	10	127.5	2.5
6	3	3:22:01	0:01:41	4	8	119.7	2.0
7	4	21:30:05	0:11:33	4	7	111.6	1.8

Multigrid preconditioner, space-time coarsening

S.-Lv.	#int	T_{opt}	T_{sim}	#NL	$\sum$#LIN	$\frac{T_{\text{opt}}}{T_{\text{sim}}}$	$\frac{\sum \#\text{LIN}}{\#\text{NL}}$
5	40	0:36:46	0:00:13	4	17	175.1	4.2
6	80	3:11:02	0:01:41	4	15	113.1	3.8
7	160	21:41:53	0:11:33	4	11	112.6	2.8

Table 2: *Driven–Cavity:* Solver statistics for distributed control. Single grid and multigrid solver applied on different coarsening strategies.

The numerical test is applied for four different solver configurations. The first part of the table contains the result for a single grid CG solver. In the second and third part, a multigrid solver is used where the space-time hierarchy is build up using coarsening in space only (until space level 4) or in time only (until time time mesh has 20 time intervals). The last part finally uses full space-time coarsening, i.e., the coarse meshes are generated by coarsening in space and time.

One can see that for this configuration, already the single-grid solver provides linear complexity. Each refinement gives a factor of eight in the number of unknowns and the computing

time. The ratio between simulation and optimisation is a factor of about 80.

If space-time multigrid is used for preconditioning (second to fourth part of the table), the results are two-fold. The total number of multigrid steps in this test is either constant or even reducing with increasing refinement level, in particular upon increasing the number of timesteps. Counting the number of CG steps on the finest level, there are 63 CG steps for the single grid solver and 28 steps (7×NSM, with NSM=4 smoothing steps per multigrid step) for the multigrid solver with time coarsening. So multigrid successfully accellerates the convergence. A hierarchy generated from pure space-coarsening is rather ineffective.

From the viewpoint of numerical efficiency, however, the overhead for the space-time Newton algorithm is rather large. The ratio $\frac{T_{\text{opt}}}{T_{\text{sim}}}$ is much higher than that of a single-grid algorithm. It depends on the configuration of the coarse grid problems and the solver parameters if the approach is effective. In the above test, one can expect that the use of the multigrid approach will pay off for large problems if space-time coarsening is used. The convergence speeds up with higher refinement levels and the effort for solving the coarse grid problems is not too large. Possible enhancements which may help to render the approach more efficient than a single-grid solver are the choice of a different hierarchy (e. g., coarsening twice in time per space coarsening), the use alternative smoothers (e. g., GMRES) or the application of an advanced strategy to choose the stopping criteria of all the involved solver components. A detailed analysis is, however, out of the scope of this work.

Comparison to SQP In the following, a small comparison of the solver efficiency results between the Newton solver in this paper and the SQP-type solver analysed in [13, 14, 18] is drawn. The latter one applies an inexact Newton strategy in the primal/dual space where the solution vector is given as $x = (y_0, p_0, \lambda_0, \xi_0, ..., y_N, p_N, \lambda_N, \xi_N)$. The control is eliminated. An outer space-time Newton solver reduces the nonlinear residual by the factor 10^{-6}. Linear subproblems are solved either with a one-level space-time BiCGStab(FBSimSolver) solver or a space-time multigrid solver (using V-cycle, four steps BiCGStab(FBSimSolver) for smoothing and BiCGStab(FBSimSolver) for coarse grid solving), cf. [18]. The stopping criterion of the linear solver is configured adaptively to obtain quadratic convergence. Subproblems in space are solved with a spatial multigrid up to two digits. The test configuration is chosen as above.

Table 3 gives the results for this solver. The solver is very stable, it basically needs only three nonlinear iterations to converge. In comparison to the Newton solver, for low levels, the computing time is rather the same. For higher levels, if a space-time multigrid preconditioner is applied, the SQP solver is more efficient in this example. Space-time multigrid is indeed necessary in this case, as a single-grid solver loses efficiency on higher levels – the number of linear steps per nonlinear step #LIN/#NL rises if only BiCGStab is applied. However, one should be careful with a comparison of the total time T_{opt} between both solvers, as gaining six digits in the primal/dual space does not necessarily mean to gain six digits in the control space and vice versa.

Remark: The number of nonlinear iterations differs whether the one-level preconditioner or the space-time multigrid preconditioner is applied. This is due to technical reasons. The BiCGStab solver checks the preconditioned residual while the multigrid preconditioner checks the real residual in the stopping criterion. As a consequence, BiCGStab does not solve accurately enough for the Newton to converge in three steps while multigrid does.

Single grid BiCGStab preconditioner

S.-Lv.	#int	T_{opt}	T_{sim}	#NL	$\sum$#LIN	$\frac{T_{\text{opt}}}{T_{\text{sim}}}$	$\frac{\sum \#\text{LIN}}{\#\text{NL}}$
5	40	00:13:58	0:00:13	5	25	66.5	5.0
6	80	01:54:21	0:01:41	5	37	67.7	7.4
7	160	18:24:56	0:11:33	4	36	95.6	9.0

Multigrid preconditioner, space-time coarsening

S.-Lv.	#int	T_{opt}	T_{sim}	#NL	$\sum$#LIN	$\frac{T_{\text{opt}}}{T_{\text{sim}}}$	$\frac{\sum \#\text{LIN}}{\#\text{NL}}$
5	40	00:15:26	0:00:13	3	6	41.5	2.0
6	80	02:04:08	0:01:41	3	7	40.3	2.3
7	160	11:24:15	0:11:33	3	6	53.9	2.0

Table 3: *Driven–Cavity:* Solver statistics for distributed control. SQP-type solver in the primal/dual space (u eliminated), space-time BiCGStab and multigrid preconditioner.

6.2 L^2 boundary control for Backward-facing step

Example We consider the optimal L^2 boundary control of the Navier–Stokes equations, see Section 2.2. The basic domain for this test is a backward-facing step geometry, see Figure 2, on a time interval $[0, T]$ with $T = 10$. On the left, a maximum inflow $y_{\max} = 1.5$ is prescribed, while on the right, do-nothing boundary conditions characterise the outflow; using $\nu = 1/100$, this results in a Re=100 optimisation. The part $\Gamma_C = \{2\} \times (0.5, 1) \subset \partial\Omega$ defines a control boundary of length 0.5 on the top of the step.

The initial flow y^0 is the fully developed nonstationary Navier–Stokes flow, the target flow is the stationary Stokes flow, restricted to the observation area $\Omega_s = [3, 4] \times [0, 1]$ (which induces the right-hand side "$y - z$" of the control equation being replaced by $\chi_{\Omega_s} \cdot (y - z)$, with χ_{Ω_s} the characteristic function of Ω_s). The regularisation parameter for the control is set to $\alpha = 0.2$. Table 4 gives an overview about the problem size; the cells on the coarse mesh have a size of $h = 1.0$. Figure 3 visualises the controlled flow at $t = 1.25$ and $t = 5.0$.

Single-grid and time-multigrid test Table 5 depicts the solver statistics for a single-level CG solver and a multigrid solver, carried out on different space-time levels. Due to the small number of unknowns in the control in space, any coarsening in space would not make much sense. Therefore, for multigrid tests, pure time coarsening is applied until a space-time coarse mesh with 20 time intervals is reached.

The solver configuration is the same as in Section 6.1. Both types of linear solvers, the single-grid CG method as well as the time-multigrid method, converge with rather level-independent convergence rates. With four CG smoothing steps per multigrid iteration, the multigrid method needs in this example about 40 CG iterations on the finest level, which is slightly less than in the one-level CG solver case. However, due the additional overhead on the coarse levels and the fact that the time mesh is rather coarse, the application of the multigrid method is not really reasonable. The total time is about twice as high as a single-grid approach as the costs for solving the coarse grid problems is as large as the costs for the iteration on the finest grid – which is typical for a multigrid approach. One would need very fine time meshes until the use of multigrid will be advantageous.

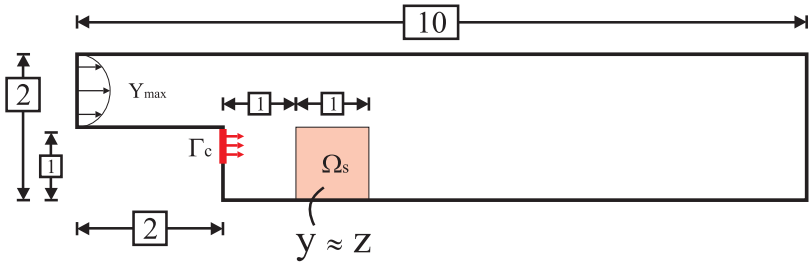

Figure 2: Test configuration 'Backward-facing step'.

S.-Lv.	#vertices	#edges	#elements	$\#\text{dof}_{pd}$	$\#\text{dof}_c$
2	97	168	72	890	1
3	337	624	288	3 362	3
4	1 249	2 400	1 152	13 058	5
5	4 801	9 408	4 608	51 458	9

Table 4: *Backward-facing step*: Mesh statistics for boundary control, different refinement levels.

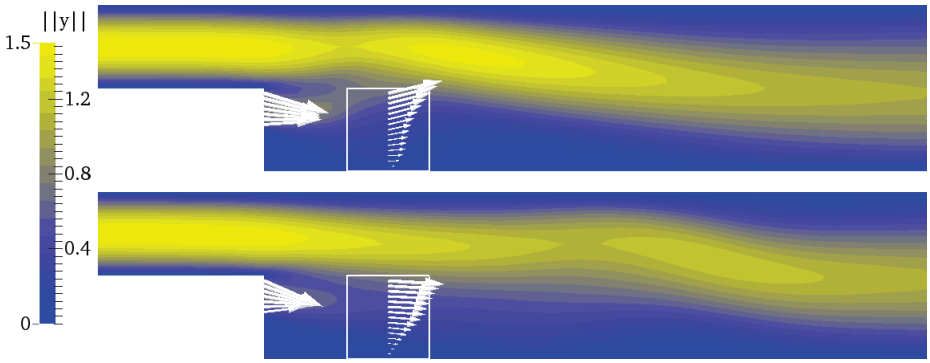

Figure 3: Test configuration 'Backward-facing step'. Controlled flow at $t = 1.25$ and $t = 5.0$.

Single-grid test

S.-Lv.	#int	T_{opt}	T_{sim}	#NL	$\sum$#LIN	$\frac{T_{\text{opt}}}{T_{\text{sim}}}$	$\frac{\sum \#LIN}{\#NL}$
3	40	0:13:46	0:00:17	5	27	48.3	5.4
4	80	2:47:18	0:02:19	6	41	72.2	6.8
5	160	23:35:08	0:14:59	6	46	94.5	7.7

Multigrid test, pure time coarsening

S.-Lv.	#int	T_{opt}	T_{sim}	#NL	$\sum$#LIN	$\frac{T_{\text{opt}}}{T_{\text{sim}}}$	$\frac{\sum \#LIN}{\#NL}$
3	40	0:32:02	0:00:17	5	8	113.7	1.6
4	80	6:18:30	0:02:19	6	10	163.9	1.7
5	160	47:50:46	0:14:59	6	10	191.6	1.7

Table 5: *Backward-facing step*: Single grid (top) and multigrid test (bottom).

7 Summary and discussion

This paper presented the application of a space-time Newton method for optimal control of the nonstationary Navier–Stokes equations. A space-time multigrid method in the control space was used for solving linear subproblems. The basic method was described and the efficiency of the method was analysed in numerical examples using distributed and L^2 boundary control.

Concerning the numerical results, it is a fact that the Newton approach does often not need multigrid for the linear subproblems to be solved. Only on very fine time meshes in combination with distributed control, the multigrid solver seems to be advantageous as the solver speeds up with the problem size.

Acknowledgements This work was financed by the program SPP1253 from the DFG, projects HI689/5-2 and TU102/24-1+2.

References

[1] R. E. Bank and T. F. Dupond. An optimal order process for solving finite element equations. *Math. Comput.*, 36(153):35–51, 1981.

[2] A. Borzi and V. Schulz. Multigrid methods for PDE optimization. *SIAM Review*, 51(2): 361–395, 2009.

[3] A. Borzi and V. Schulz. *Computational Optimization of Systems Governed by Partial Differential Equations*. SIAM, 2011.

[4] G. Büttner. *Ein Mehrgitterverfahren zur optimalen Steuerung parabolischer Probleme*. PhD thesis, Fakultät II – Mathematik und Naturwissenschaften der Technischen Universität Berlin, 2004. `http://edocs.tu-berlin.de/diss/2004/buettner_guido.pdf`.

[5] M. D. Gunzburger. *Perspectives in Flow Control and Optimization*. SIAM, 2003. ISBN 089871527X.

[6] W. Hackbusch. Fast solution of elliptic control problems. *J. Opt. Theory and Appl.*, 31 (4):565–581, 1980.

[7] W. Hackbusch. Die schnelle Auflösung der Fredholmschen Integralgleichung zweiter Art. *Beiträge zur numerischen Mathematik*, 9, 1981.

[8] W. Hackbusch. Numerical solution of linear and nonlinear parabolic optimal control problems. *Lecture Notes in Control and Information Science*, 30, 1981.

[9] W. Hackbusch. *Multi-Grid Methods and Applications*. Springer Series in Computational Mathematics. Springer, Berlin, 1985. ISBN 3-540-12761-5.

[10] M. Hintermüller and M. Hinze. A SQP-semi-smooth Newton-type algorithm applied to control of the instationary Navier–Stokes system subject to control constraints. *SIAM J. Optim.*, 16:1177–1200, 2006.

[11] M. Hinze. *Optimal and instantaneous control of the instationary Navier–Stokes equations*. Habilitation thesis, Institut für Numerische Mathematik, Technische Universität Dresden, 2000.

[12] M. Hinze and M. Vierling. The semi-smooth newton method for variationally discretized control constrained elliptic optimal control problems; implementation, convergence and globalization. *Opt. Meth. Software*, 27(6), 2012.

[13] M. Hinze, M. Köster, and S. Turek. A hierarchical space-time solver for distributed control of the Stokes equation. Preprint SPP1253-16-01, SPP1253, 2008.

[14] M. Hinze, M. Köster, and S. Turek. A space-time multigrid solver for distributed control of the time-dependent Navier–Stokes system. Preprint SPP1253-16-02, SPP1253, 2008.

[15] M. Hinze, M. Köster, and S. Turek. A hierarchical space-time solver for optimal distributed control of fluid flow, 2009. Proceedings of the Conference on Modeling, Simulation and Optimization of Complex Processes, Heidelberg, July 21-25, 2008, accepted.

[16] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich. *Optimization with PDE Constraints*, Volume 23 of *Mathematical Modelling: Theory and Applications*. Springer, Berlin, 2009. ISBN 9781402088384.

[17] V. John and G. Matthies. Higher order finite element discretizations in a benchmark problem for incompressible flows. *Int. J. Num. Meth. Fluids*, 37:885–903, 2001.

[18] M. Köster. *A Hierarchical Flow Solver for Optimisation with PDE Constraints*. Phd thesis, TU Dortmund, Lehrstuhl III für Angewandte Mathematik und Numerik, 2011. Slightly corrected version with an additional appendix concerning prolongation/restriction.

[19] M. Ulbrich. Semismooth Newton methods for the operator equations in function spaces. *SIAM J. Optim.*, 3:805–841, 2003.

[20] M. Ulbrich. Constrained optimal control of Navier–Stokes flow by semismooth Newton methods. *Syst. Contr. Lett.*, 48:297–311, 2003.

[21] H. Yserentant. Old and new convergence proofs for multigrid methods. *Acta Numerica*, pages 1–44, 1992.