Multigrid-based Solution Methods for the Optimal Control of Nonstationary Flows

Köster, M.[1]; Hinze, M.[2.] ; Turek, S.[1]
[1]Institute of Applied Mathematics, TU Dortmund
[2]Department of Mathematics, University of Hamburg
michael.koester@mathematik.tu-dortmund.de

While simulation methods in the field of Computational Fluid Dynamics (CFD) are already quite advanced, efficient and well developed numerical approaches for optimisation and optimal control in CFD are still under development. Particular challenges in this context are optimisation problems with instationary flows, resulting in discrete nonlinear systems of equations with an extremely high number of unknowns. We give a short overview about current developments in the field of solvers for such situations. Our emphasis will be placed on special space-time multigrid based solution strategies which are able to solve the underlying nonlinear system of equations with linear complexity.

1 Introduction

The optimal control of incompressible, nonstationary flow problems belongs to today's most challenging problems in the field of optimisation. Modeling leads to minimisation problems whose necessary first order optimality conditions form the so-called Karush-Kuhn-Tucker (KKT-) systems. These is the starting point for numerical solution approches. All variables in the KKT system are fully coupled, so that its discretisation leads to a very high-dimensional nonlinear system of equations.

There are different discretisation and solver approaches available to tackle KKT systems numerically, and a fairly good overview can be found in [1] and [2]. Newer works (cf. [3], [5]) start to use multigrid methods to prevent performance degeneration for fine discretisations. The trick is to apply multigrid methods simultaneously in space and time. We will sketch this approach for the optimal distributed control of the nonstationary Navier-Stokes equations.

2 Distributed flow control

A typical model problem for optimisation in CFD is optimal distributed control of the nonstationary Navier-Stokes equations, which is modelled as follows. The spatial domain is denoted by $\Omega \subset \Re^d$ (d=2,3), with boundary Γ. For T>0, $Q:=(0,T)\times\Omega$ defines a space-time cylinder and $\Sigma:=(0,T)\times\Gamma$ its space-time boundary. With $\alpha>0$ a regularisation parameter, $z:Q\to\Re^d$ a given 'target' function and y^0: $\Re^d\to\Re^d$ a given initial condition, we consider the minimisation problem

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

such that

$$y_t-\Delta y+y\nabla y+\nabla p=u \qquad \text{in Q}, \qquad (2a)$$
$$-\nabla\cdot y=0 \qquad \text{in Q}, \qquad (2b)$$
$$y=0 \qquad \text{on } \Sigma, \qquad (2c)$$
$$y(0)=y^0 \qquad \text{in } \Omega, \qquad (2d)$$

where $\|\cdot\|_{L^2(Q)}$ denotes the L_2-norm over the space-time cylinder Q. The aim of this problem is to find an optimal control u such that the flow y is as close as possible to the given target flow z in the mean-square sense.

2.1 The KKT system

Using the Lagrange multiplier approach [3], the necessary optimality conditions of the above minimization can be formulated as a nonlinear system of equations:

- State, adjoint and control equation:

$$y_t-\Delta y+y\nabla y+\nabla p=u \qquad \text{in Q}, \qquad (3a)$$
$$-\nabla\cdot y=0 \qquad \text{in Q}, \qquad (3b)$$
$$-\lambda_t-\Delta\lambda-y\nabla\lambda+(\nabla y)^T\lambda+\nabla\xi=(y-z) \quad \text{in Q}, \qquad (4a)$$
$$-\nabla\cdot\lambda=0 \qquad \text{in Q}, \qquad (4b)$$
$$\alpha u+\lambda=0 \qquad \text{in Q} \qquad (5)$$

- Boundary and initial/end time conditions

$$y = \lambda = 0 \qquad \text{on } \Sigma,$$
$$y(0) = y^0, \quad \lambda(T) = 0 \qquad \text{in } \Omega,$$

In this system, $\lambda:Q\to\Re^d$ denotes the so-called adjoint velocity and $\xi:Q\to\Re$ the adjoint pressure. Equation (3) proceeds forward in time, equation (4) backward in time and the control equation (5) couples the forward and backward equation. Therefore, the system represents a fully coupled boundary value problem in space and time.

2.2 Space-time discretisation and hierarchy

For the discretisation in space, we choose the finite element space Q_2 of quadratic finite elements on quadritalterals for the velocities and controls. The pressures are discretised using P_1^{disc}, the space of piecewise discontinuous linear finite elements. Time discretisation is carried out, e.g., with the implicit Euler scheme. Higher-order schemes like Crank-Nicolson are also possible, but lead to more complicated numerical algorithms (see [3]).

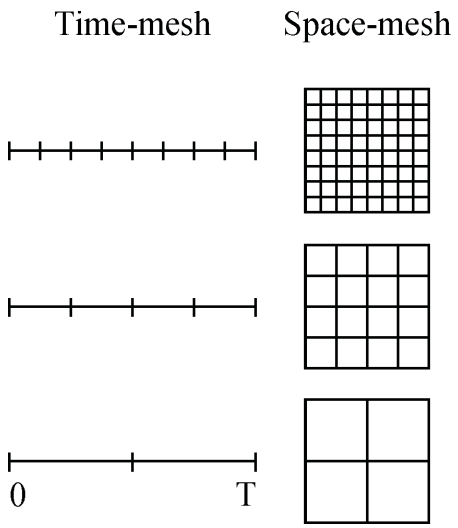

Figure 1: Hierarchy of three space-time meshes. Coarse mesh (level 1) on the bottom.

A space-time mesh hierarchy of $L\in\aleph$ levels is obtained as follows:

1. Spatial discretisation: Let $T_1,\ldots,T_L$ define a hierarchy of meshes in space generated by regular refinement of a basic mesh. On each level, a finite element discretisation is carried out, resulting in a hierarchy of finite element spaces $V_1,\ldots,V_L$ for velocity/control and $Z_1,\ldots,Z_L$ for the pressure.

2. Time discretisation: The time interval [0,T] is subdivided into N time intervals of equal length k=T/N with gridpoints $t_0=0 < t_1 < \ldots < t_N=T$. A time hierarchy is generated by bisection, so that time level l contains $2^{l-1}N$ time intervals.

The space-time space W_l, l=1,…,L, is defined by the combination of (V_l,Z_l) in space and $2^{l-1}N$ intervals in time. Figure 1 sketches a hierarchy of three space-time levels. On the finest space W_L, a solution to the optimal control problem (1) shall be computed.

3 Solver design

Different solver approaches have been developed in recent years to tackle the above KKT system (3)-(5) numerically, see [1], [3], [4], [5]. In the following, we sketch two types of solvers which couple Newton techniques for the nonlinear parts with multigrid techniques for the linear subproblems.

3.1 The Newton approach

The Newton approach is based on equation (5) whichwe write in operator form as

$$F(u) := \alpha u + \lambda = 0 \qquad \text{in Q.} \qquad (6)$$

In this equation, $\lambda=\lambda(y(u))$ is a solution of the adjoint equation (4) and $y=y(u)$ a solution of the primal equation (3). Given an initial iterate u_0, the Newton iteration for finding the solution of (6) reads

$$u_{n+1} = u_n - F'(u_n)^{-1} F(u_n), \qquad (7)$$

which consists of three steps:

1. Form the residual

$$d_n = -F(u_n). \qquad (8a)$$

This necessitates three substeps:

- One forward iteration to solve (3) with given u_n; this provides an intermediate primal velocity y_n and pressure p_n.
- One backward iteration to solve (4) with given y_n; this results in an intermediate adjoint velocity λ_n and pressure ξ_n.
- Form

$$d_n = -(\alpha u_n + \lambda_n).$$

2. Calculate $\bar{u}$ by solving the linear equation:

$$F'(u_n)\bar{u} = \alpha\bar{u} + \bar{\lambda} = d_n \qquad (8b)$$

3. Update the solution:

$$u_{n+1} = u_n + \bar{u} \tag{8c}$$

In step (8b), $\bar{\lambda} = \bar{\lambda}(\bar{y}(\bar{u}))$ is the solution of the adjoint equation

$$\begin{aligned} -\bar{\lambda}_t - \Delta\bar{\lambda} - y\nabla\bar{\lambda} + (\nabla y)^T\bar{\lambda} + \nabla\bar{\xi} & \\ = \bar{y} - (\nabla\bar{y})^T\lambda + \bar{y}\nabla\lambda & \quad \text{in Q,} \quad (9a) \\ -\nabla\cdot\bar{\lambda} = 0 & \quad \text{in Q,} \quad (9b) \\ \bar{\lambda} = 0 & \quad \text{on } \Sigma, \quad (9c) \\ \bar{\lambda}(T) = 0 & \quad \text{on } \Omega, \quad (9d) \end{aligned}$$

with $\bar{y} = \bar{y}(\bar{u})$ the solution of the linearised primal equation

$$\begin{aligned} \bar{y}_t - \Delta\bar{y} + y\nabla\bar{y} + \bar{y}\nabla y + \nabla\bar{p} = \bar{u} & \quad \text{in Q,} \quad (10a) \\ -\nabla\cdot\bar{y} = 0 & \quad \text{in Q,} \quad (10b) \\ \bar{y} = 0 & \quad \text{on } \Sigma, \quad (10c) \\ \bar{y}(0) = 0 & \quad \text{in } \Omega. \quad (10d) \end{aligned}$$

Step (8b) is numerically expensive as a linear system in space and time has to be solved. This can be done iteratively by a multigrid approach.

The linear solver in the Newton approach

For the linear equation (8b), an arbitrary iterative algorithm based on defect corrections can be used. For example, given a damping parameter ω>0 and an initial guess $\bar{u}_0$ on level L, a simple Richardson iteration reads

$$\bar{u}_{m+1} = \bar{u}_m + \omega(u_n - F'(u_n)\bar{u}_m). \tag{11}$$

Similarly, it is also possible to apply a BiCGStab or GMRES algorithm. All such iterative algorithms produce a sequence of iterates $\bar{u}_m$ that converge to the solution $\bar{u}$ of (8b), which is a function on the space-time cylinder. One has to note here that the creation of the defect is the expensive part: To calculate the matrix-vector product $F(u_n)\bar{u}_m$, the following three steps have to be done:

- Solve the primal equation (10); this is a simulation of a linear equation forward in time and computes an auxiliary $\bar{y}_m$ and $\bar{p}_m$.
- Solve the adjoint equation (9); this is a simulation of a linear equation backward in time and computes an auxiliary $\bar{\lambda}_m$ and $\bar{\xi}_m$.
- In each timestep, apply the linear combination

$$F(u_n)\bar{u}_m = \alpha\bar{u}_m + \bar{\lambda}_m. \tag{12}$$

Multigrid in the Newton approach

Multigrid is a special type of iterative algorithm and can be embedded in the above iterative structure as well. It builds upon the hierarchy $W_1,\ldots,W_L$ of spaces in space and time. The basic idea reads as follows:

1. Take an initial guess $\bar{u}_0$ as before.

2. Smoothing: On level L, carry out a fixed number of iterations with an iterative algorithm (CG, GMRES) as above, e.g., NSM=4 steps. This gives an intermediate solution $\bar{u}_m^{NSM}$.

3. Coarse grid correction: On level L-1, use $\bar{u}_m^{NSM}$ to calculate a 'coarse grid solution' $\tilde{u}$ to $\bar{u}$. Combine $\tilde{u}$ and $\bar{u}_m^{NSM}$ to get a new approximate $\bar{u}_{m+1}$. More precisely, one applies the defect correction formula

$$\bar{u}_{m+1} := \bar{u}_m^{NSM} + P(F'_{L-1}(u_n))^{-1} R(u_n - F'(u_n)\bar{u}_m^{NSM})$$

with R a suitable restriction operator from level L to L-1, P a prolongation operator from level L-1 to L and $F'_{L-1}(u_n)$ an approximation to the operator $F'(u_n)$ on level L-1.

This process can be carried out recursively: The approximate solution on level L-1 can be calculated by a couple of smoothing steps on level L-1 followed by a coarse grid correction from level L-2, etc. On level l=1, the iterative algorithm (CG, GMRES,…) has to be carried out until convergence to calculate a basic coarse grid solution. A good overview about multigrid can be found in [6] and [7].

3.2 The SQP approach

The above Newton approach can be interpreted as an elimination of the equations (3) and (4) from the KKT system. An iteration is carried out only in based on (5), using (3) and (4) as auxiliary problems. The SQP approach is an alternative approach which does the other way around, formulating an iteration in y, p, λ, ξ and u simultaneously. It is usually more effective than the Newton approach. However, the approach is very intrucive and requires a certain amount of coding, as it cannot be realised with black-box solvers.

We consider a special SQP approach which eliminates the control. Elimination of equation (5) leads to the following set of equations:

- State equation:

$$y_t - \Delta y + y\nabla y + \nabla p = -\frac{1}{\alpha}\lambda \qquad \text{in Q,} \qquad (13a)$$

$$-\nabla \cdot y = 0 \qquad \text{in Q,} \qquad (13b)$$

- Adjoint equation:

$$-\lambda_t - \Delta\lambda - y\nabla\lambda + (\nabla y)^T \lambda + \nabla\xi = (y - z) \qquad \text{in Q,} \qquad (14a)$$

$$-\nabla \cdot \lambda = 0 \qquad \text{in Q,} \qquad (14b)$$

- Boundary conditions, initial/end time conditions (see above).

This equation can be expressed in operator form by

$$G(y, p, \lambda, \xi) = (0,0,0,0) \qquad (15)$$

with $G=(G_1,G_2,G_3,G_4)$, $G_i=G_i\ (y,p,\lambda,\xi)$ the defects of (13) and (14) given by

$$G_1 = y_t - \Delta y + y\nabla y + \nabla p + \frac{1}{\alpha}\lambda$$

$$G_2 = -\nabla \cdot y$$

$$G_3 = -\lambda_t - \Delta\lambda - y\nabla\lambda + (\nabla y)^T \lambda + \nabla\xi - y + z$$

$$G_4 = -\nabla \cdot \lambda$$

Given an initial iterate $x_0:=(y_0,p_0,\lambda_0,\xi_0)$, a Newton iteration has now to be applied to (15). With $x_n:=(y_n,p_n,\lambda_n,\xi_n)$, this reads

$$x_{n+1} = x_n - G'(x_n)G(x_n)\,. \qquad (16)$$

Again, this can be decomposed into three steps:

1. Create the residual

$$d_n = -G(x_n)\,. \qquad (17a)$$

2. Solve the linear equation

$$G'(x_n)\bar{x} = d_n\,. \qquad (17b)$$

3. Update the solution by

$$x_{n+1} = x_n + \bar{x}\,. \qquad (17c)$$

Here, (17b) means that $\bar{x} = (\bar{y}, \bar{p}, \bar{\lambda}, \bar{\xi})$ is a solution of the equations (9) and (10), respectively, with $y=y_n$, and $\lambda=\lambda_n$.

The defect

In contrast to the Newton approach, the creation of the residual as well as the solution of the linear equation is fundamentally different. The creation of the residual in (17a) is rather cheap. For a given iterate x_n, no system has to be solved. Instead, the operator has to be *applied* to x_n, which means a couple of matrix-vector products (for the Laplace/gradient/divergence operators) or an assembly (for the nonlinearities), respectively, in each timestep.

The linear equation

The main difficulty in this approach is the linear system (17b) which has to be solved for all primal/adjoint variables simultaneously. This system has a very specific form. For example, if the implicit Euler scheme is used for the time discretisation, one can write the discrete solution vector containing all timesteps in the form

$$\vec{x}_n = (x_n^0, x_n^1, ..., x_n^N)$$

with $x_n^i = (y_n^i, p_n^i, \lambda_n^i, \xi_n^i)$ the solution at t_i, i=0,…,N. A matrix-vector product like in (17b) is then carried out with the discrete counterpart of $G'(x_n)$, given by a (block) tridiagonal matrix of the form

$$\mathbf{G}'(\vec{x}_n) = \begin{pmatrix} A_0 & M_d & & & \\ M_p & A_1 & M_d & & \\ & M_p & A_1 & \ddots & \\ & & \ddots & \ddots & \\ & & & M_p & A_N \end{pmatrix}$$

The matrices M_p are weighted mass matrices, introduced by the implicit Euler scheme in the primal equation. Similarly, the matrices M_d are weighted mass matrices from the adjoint equation. The diagonal blocks A_i contain for each timestep a (linearised) Navier-Stokes operator that represents the coupled equations (13) and (14), as well as weighted mass matrix from the timestepping scheme.

The linear solver in the SQP approach

As described above, applying the operator $G'(x_n)$ corresponds in the discrete world to a matrix-vector product with a (block) tridiagonal matrix. Therefore, it is possible to use any iterative algorithm for linear systems (BiCGStab, GMRES) in order to solve (17b). For example, given an initial iterate

$\bar{x}_0 := (\bar{y}_0, \bar{p}_0, \bar{\lambda}_0, \bar{\xi}_0)$ and a damping parameter ω>0, a Richardson iteration reads

$$\bar{x}_{m+1} = \bar{x}_m + \omega(d_n - G'(x_n)\bar{x}_m). \qquad (18)$$

Provided ω is small enough, this should to converge for m→∞ to a solution $\bar{x}$ of (17b). Using a BiCGStab or GMRES algorithm however is expected to converge much more rapidly.

Preconditioning

A crucial point concerning the SQP approach is that one can easily derive efficient preconditioners. For example, the preconditioned version of the Richardson iteration above reads

$$\bar{x}_{m+1} = \bar{x}_m + \omega C^{-1}(d_n - G'(x_n)\bar{x}_m) \qquad (19)$$

with C an approximation to $G'(x_n)$. Due to the fact that $G'(x_n)$ can be represented as a tridiagonal matrix, straightforward and efficient preconditioners are the Block-Jacobi method, realised by

$$\mathbf{C}_J = \begin{pmatrix} A_0 & & & & \\ & A_1 & & & \\ & & A_2 & & \\ & & & \ddots & \\ & & & & A_N \end{pmatrix},$$

the forward Block Gauß-Seidel preconditioner, realised by

$$\mathbf{C}_{GS}^{f} = \begin{pmatrix} A_0 & & & & \\ M_p & A_1 & & & \\ & M_p & A_2 & & \\ & & \ddots & \ddots & \\ & & & M_p & A_N \end{pmatrix},$$

the backward Block Gauß-Seidel C_{GS}^{b} (realised by using M_d on the upper diagonal instead of M_p on the lower one) or combinations of these. Applying C^{-1} that way reduces to the application of the A_i^{-1} in all timesteps i=1,…,N. These problems, however, are problems in space, and efficient monolithic spatial multigrid solvers are available.

Monolithic Multigrid in the SQP approach

For an efficient application of a monolithic multigrid strategy, one has to carry out the discretisation of the KKT system (i.e., equation (15)) on all levels 1,…,L. Multigrid is used to solve equation (17b) as a special type of an iterative solution algorithm. The basic idea reads as follows, similar to the Newton approach.

1. Take an initial guess $\bar{x}_0$ on level L as before.

2. Smoothing: On level L, carry out a fixed number of iterations with an iterative algorithm (BiCGStab, GMRES) as above, e.g., NSM=4 steps. This gives an intermediate solution $\bar{x}_m^{NSM}$.

3. Coarse grid correction: On level L-1, use $\bar{x}_m^{NSM}$ to calculate a 'coarse grid solution' $\tilde{x}$ to $\bar{x}$. Combine $\tilde{x}$ and $\bar{x}_m^{NSM}$ to get a new approximate $\bar{x}_{m+1}$. More precisely, one applies the defect correction formula

$$\bar{x}_{m+1} := \bar{x}_m^{NSM} + P(G'_{L-1}(u_n))^{-1} R(d_n - G'(x_n)\bar{x}_m^{NSM})$$

with R a suitable restriction operator from level L to L-1, P a prolongation operator from level L-1 to L and $G'_{L-1}(u_n)$ an approximate solution operator for $G'(u_n)$ on level L-1.

Again, the computation of the coarse grid correction $\tilde{x}$ can iteratively be carried out using a smoothing/ coarse grid correction strategy that involves level L-2, L-3, etc. On level l=1, a coarse grid solver has to be used, i.e., the iterative algorithm has to iterate until convergence.

4 Numerical examples

In this section, we present some numerical exmples to demonstrate the applicability of the proposed solver approaches. We consider the following test example, based on the Flow-Around-Cylinder benchmark problem from [8], see also [9].

The spatial domain is described by a rectangle without an inner cylinder,

$$\Omega := (0,2.2)\times(0,0.41)\setminus B_r(0.2,0.2), \quad r := 0.05$$

The boundary of this domain is decomposed into five parts:

$$\Gamma_1 := \{0\}\times[0,0.41], \quad \Gamma_2 := (0,2.2]\times\{0\},$$
$$\Gamma_3 := \{2.2\}\times(0,0.41), \quad \Gamma_4 := \{0,2.2]\times\{0.41\},$$
$$\Gamma_5 = \partial B_r(0.2,0.2).$$

Boundary conditions are defined as y(x,t):=(0,0) for $x \in \Gamma_2 \cup \Gamma_4 \cup \Gamma_5$, do-nothing boundary conditions on

Γ_3 and a parabolic profile with maximum inflow velocity Y_{max}=1.5 at Γ_1. Using ν=1/1000, this results in a Re=100 optimisation. The time interval for this test case is [0,T] with T=0.35 which roughly corresponds to one oscillation in the uncontrolled flow.

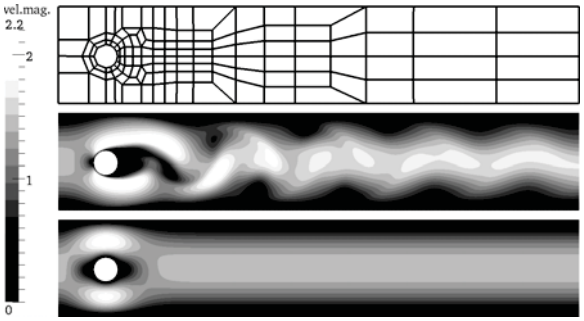

Figure 2: Mesh (top), y^0 (centre) and z (bottom).

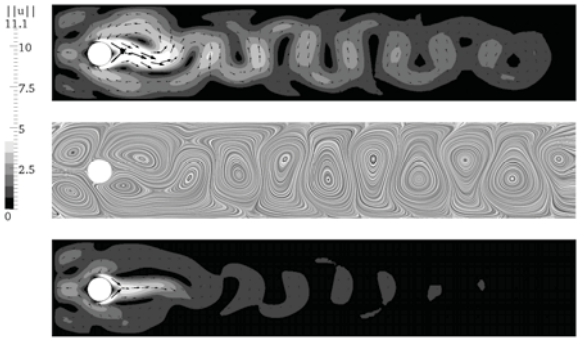

Figure 3: u at t=0.05 (top) and corresponding SurfaceLIC representation (centre). Bottom: u at t=0.5.

The initial flow y^0 is the fully developed nonstationary Navier-Stokes flow, the target flow z is the stationary Stokes flow. Figure 2 depicts the mesh, the initial condition and the target flow, Figure 3 the control at different points in time. Table 1 gives an overview about the problem size.

Space-Lev.	#vertices	#edges	#elements
1	156	286	130
2	572	1 092	520
3	2 184	4 264	2 080
4	8 528	16 848	8 320

Table 1: Mesh size for different refinement levels in space.

4.1 Newton solver

For the following tests, the described space-time Newton algorithm preconditioned with a space-time multigrid solver in the control space is chosen. The space-time Newton algorithm was configured to reduce the initial residual by six digits. The space-time multigrid algorithm used in every Newton step reduced its initial residual by two digits using a V-cycle. For smoothing, four steps of a space-time CG method are applied. The coarse grid solver is a CG solver as well which damps its initial residual by two digits. During the calculation of the nonlinear and linear residuals, forward and backward iterations have to be performed. Their local systems in space are solved such that the residual drops below 10^{-14}.

Single-grid preconditioning

The first test considers a simple one-level solver configuration, see Table 2. On different refinement levels in space and time the Newton algorithm with embedded CG solver is applied. We use the following notations:

- 'SLv' = refinement level in space
- 'N' = number of intervals in time
- T_{opt} = time for the the optimisation.
- T_{sim} = time needed for the computation of the first forward simulation
- #NL = number of Newton iterations
- Σlin = sum of all iterations of the linear solver

The number of CG iterations increases slightly with increasing the number of timesteps, so level-independent convergence is nearly lost. The refinement in space does not seem to be relevant for the efficiency of the CG solver. CG only needs about 7-10 iterations to reduce the residual by a factor of 10^{-2} in every Newton iteration, which leads in this example to a ratio of about 30-50 between the simulation and the optimisation.

SLv	N	T_{opt}	T_{sim}	#NL	Σlin	$\frac{\#NL}{\Sigma CG}$	$\frac{T_{opt}}{T_{sim}}$
3	40	4:12:29	6:38	5	35	7.0	38.1
3	80	9:01:58	12:17	5	43	8.6	44.1
3	160	18:38:29	23:56	5	47	9.4	46.7
2	40	42:55	1:09	5	36	7.2	37.1
3	40	4:12:29	6:38	5	35	7.0	38.1
4	40	15:36:08	25:48	5	36	7.2	36.3
2	20	20:33	0:40	5	32	6.4	31.1
3	40	4:12:29	6:38	5	35	7.0	38.1

Table 2: Newton with single-grid CG solver

Multigrid preconditioning

Table 3 depicts results for different levels using a multigrid solver for the linear subproblems. For the coarse mesh, space level two with 20 timesteps is chosen.

The total number of multigrid iterations is rather constant or even reducing for higher levels, which is an advantage to the one-level solver. However, multigrid needs about 1.5-2.5 iterations per nonlinear iteration, which corresponds to 6-10 CG iterations on the finest level (due to four smoothing steps per multigrid iteration). This is rather similar to the one-level solver test. The ratio between optimisation and simulation is with a factor of 50 higher than in the one-level case. So the additional overhead will pay off only for meshes with a finer mesh resolution in time.

SLv	N	T_{opt}	T_{sim}	#NL	Σlin	$\frac{\#NL}{\Sigma CG}$	$\frac{T_{opt}}{T_{sim}}$
3	40	5:40:00	6:38	4	8	2.0	51.3
4	80	46:03:22	52:24	5	9	1.8	52.7
5	160	297:26:50	6:13:18	5	8	1.6	47.8

Table 3: Newton with multigrid solver

4.2 SQP solver

In this section, the same example is calculated using the SQP approach. The basic test settings are the same. However, it is possible to use different solver settings here as the SQP approach allows solving subproblems inexactly without destroying the solution.

a) The outer Newton solver is configured to reduce the nonlinear residual by six digits which is similar to the configuration of the Newton above

b) For the inner linear solver, we apply on the one hand a single-grid space-time BiCGStab solver, on the other hand a space-time multigrid solver. Both reduce the residual by two digits. If a space-time multigrid solver is used, the coarse grid solver and the smoother is BiCGStab with a forward-backward simulation used for preconditioning. The smoother applies four (post-) smoothing steps.

c) For the solver in space in each time interval, we apply a spatial monolithic multigrid algorithm which reduces its initial residual by two digits. Smoothing is carried out with a local pressure-Schur-Complement ('VANKA'-) like technique.

Single-grid preconditioning

Table 4 shows the results using the single-grid BiCGStab solver. The time T_{sim} for the simulation was taken from the Newton approach in the control space, see Table 2.

The number of nonlinear iterations is independent of the level. The number of iterations of the linear solver is depending on the time interval length, it rises slightly if the number of time intervals increases. The refinement level in space does not seem to have an influence to the solver efficiency. This behaviour has also been seen in the test for the Newton approach in the control space.

Comparing optimisation and simulation, the optimisation only needs about 10-20 times longer than the simulation. So in this example, the control-eliminated SQP approach seems to be more efficient than the Newton approach in the control space, compare Table 2.

SLv	N	T_{opt}	T_{sim}	#NL	Σlin	$\frac{\#NL}{\Sigma CG}$	$\frac{T_{opt}}{T_{sim}}$
3	40	1:53:48	6:38	6	31	5.2	17.2
3	80	2:16:00	12:17	6	31	5.2	11.1
3	160	4:40:35	23:56	6	33	5.5	11.7
2	40	16:12	1:09	6	32	5.3	14.0
3	40	1:53:48	6:38	6	31	5.2	17.2
4	40	7:28:09	25:48	5	26	5.2	17.4
2	20	9:05	0:40	5	25	5.0	13.8
3	40	1:53:48	6:38	6	31	5.2	17.2
4	80	12:09:20	52:19	6	34	5.7	13.9

Table 4: SQP with single-grid solver

Multigrid preconditioning

Table 5 depicts the results in the case where a multigrid solver is used for preconditioning in space and time.

The convergence behaviour of the multigrid-based approach does not depend on the refinement level in contrast to the single-grid approach. Concerning the absolute time and the ratio between optimisation and simulation, this approach is slightly more expensive than the single-grid approach. One can see a factor of about 20. In total, the complete approach is still only half as expensive in this configuration compared to the Newton approach in the control space, see Table 2 and Table 3.

SLv	N	T_{opt}	T_{sim}	#NL	Σlin	$\frac{\#NL}{\Sigma CG}$	$\frac{T_{opt}}{T_{sim}}$
3	40	2:22:30	6:38	6	9	1.5	21.5
4	80	16:41:27	52:24	6	10	1.7	19.1

Table 5: SQP with multigrid solver

5 Conclusions and Outlook

We presented two modern approaches for the optimal control of nonstationary, incompressible flows. The methods are rather general and can in a similar way also be applied to other types of problems. It can be said that the complete approach can also be extended to boundary control problems. Furthermore, bounds in the control are usually also possible, cf. [3].

The basic idea is to derive a KKT system which models the first order necessary optimality conditions of a given minimisation problem. Depending on the approach, some equations are eliminated, and a Newton solver is applied to the remaining equations. The linear system inside of the Newton is solved with a space-time multigrid approach.

The standard Newton approach in the control space has the advantage that it can be built upon an existing flow solver, as it only needs a forward and a backward solver for the primal/adjoint equations. The SQP approach on the other hand is more intrusive, but (due to more freedom in the choice of the stopping criteria) also more efficient.

In both cases, the multigrid approach leads to level-independent convergence rates. This means that the effort necessary for solving the optimisation problem grows linearly with the problem size, which is optimal. However, the additional overhead renders this approach only useful for very fine discretisations, in particular, with a lot of timesteps.

There are even more methods available, which can be seen as modifications of the above two. In the ‘reduced SQP’ approach for example, one combines the above two approaches (see [1]). Furthermore, special BFGS-based variants are also available which are applicable, e.g., for boundary control problems with only few unknowns. Numerical results for these approaches are, however, not yet available.

6 References

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

[2] 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.

[3] M. Köster. A Hierarchical Flow Solver for Optimisation with PDE Constraints. Phd thesis, TU Dortmund, Lehrstuhl III für Angewandte Mathematik und Numerik, 2011.

[4] 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.

[5] 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.

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

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

[8] S. Turek and M. Schäfer. Benchmark computations of laminar flow around cylinder. In E.H. Hirschel, editor, Flow Simulation with High-Performance Computers II, Volume 52 of Notes on Numerical Fluid Mechanics, pages 547-566. Vieweg, 1996.

[9] M. Hinze and K. Kunisch. Three control methods for time-dependent fluid flow. Flow Turbulence Combust., 68:273–298, 2000.